Future Generation Computer Systems 161 (2024) 415–431

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

IoT-driven blockchain to manage the healthcare supply chain and protect medical records

Alessandra Rizzardi, Sabrina Sicari [*], Jesus F. Cevallos M., Alberto Coen-Porisini

Dipartimento di Scienze Teoriche e Applicate, Università degli Studi dell'Insubria, via O. Rossi 9, 21100 Varese, Italy

ARTICLE INFO

Keywords:
Blockchain
Internet of Things
Access control
Security policies
Smart health

ABSTRACT

Healthcare supply chain domain and medical records' management face numerous challenges that come with new demands, such as customer dissatisfaction, rising healthcare costs, tracking and traceability of drugs, and security and privacy related to the sensitive information managed in such a domain. Executing processes related to healthcare domain in a trusted, secure, efficient, accessible and traceable manner is challenging due to the fragmented nature of the healthcare supply chain, which is prone to systemic errors and redundant efforts that may compromise patient safety and negatively impact health outcomes. To cope with such issues, blockchain technology, combined with the Internet of Things (IoT), can offers a reliable way to track and trace products and protect medical data though a peer-to-peer distributed, secure, and shared ledger. Hence, this paper proposes an IoT-driven blockchain-based architecture to manage the healthcare supply chain and protect medical records from tampering and access violation. Hyperledger Fabric, which is a permissioned blockchain, has been adopted due to the sensitive and private nature of the collected data. The envisioned network has been implemented and performance has been evaluated in terms of execution time, resources consumption and throughput.

1. Introduction

In traditional business, social, and political systems, various agreements, contracts, and financial transactions are maintained and recorded in a fixed structure. With the rapid growth of internet technologies and digitization, a world, where transparency is expected by end-users, is becoming even more prevalent. In today's digital era, stakeholders in business and other communication fields want to transact without intermediaries and expect trust and reliability through technology design. The spreading of healthcare 4.0 systems and technologies is a matter of fact [1], along with the identified challenges, such as: (i) network latency; (ii) fragmented and erroneous health data; (iii) no provision for real-time remote health data collection, analysis, diagnosis, predicting critical conditions, and suggesting treatment measures; (iv) gaps in workflows due to incompatible and vendor-specific healthcare solutions; (v) lack of data privacy, transparency, and integrity; (vi) no control by the users over who has access to their health data; (vii) lack of complete and comprehensive health history of patients; (viii) lack of a trusted, secure health data-sharing platform to process data gathered from different healthcare systems; (ix) health data of patients often stored in a local central database which may not be fault-tolerant.

In such a context, the blockchain technology has gained substantial recognition for its ability to induce transformation and innovation in existing business models and frameworks. Blockchain technology has been highly appreciated for its decentralized and peer-to-peer communication. The emergence of blockchain technology has had a tremendous impact on the business and IT industries. In recent years, large companies, such as IBM,[1] have made efforts to provide more powerful, reliable, and cost-efficient platforms for it. With features such as scalability, programmability, the optimized data structure for blocks and transactions, and new consensus methods, there is now a huge demand for blockchain technology in real-world applications. Hence, the application of such a technology to the management domain and its processes has attracted increasing interest from both academia and industry. Indeed, there are a lot of different scenarios in which this technology can be adopted, that spaces from transport and logistics fields [2], to finance and banking applications [3], agriculture and food supply chains [4], and also there are studies for the application of this technology in the healthcare [5]. There are so many possible

[*] Corresponding author.
E-mail addresses: alessandra.rizzardi@uninsubria.it (A. Rizzardi), sabrina.sicari@uninsubria.it (S. Sicari), jf.cevallosmoreno@uninsubria.it (J.F. Cevallos M.), alberto.coenporisini@uninsubria.it (A. Coen-Porisini).

[1] IBM Blockchain, https://www.ibm.com/blockchain.

https://doi.org/10.1016/j.future.2024.07.039
Received 9 April 2024; Received in revised form 11 July 2024; Accepted 20 July 2024
Available online 23 July 2024

application fields for this technology mainly because the blockchain can provide traceability and transparency [6].

However, this kind of domains faces serious limitations and challenges. In fact, the blockchain technology represents a continuously growing tamper-resistant ledger that maintains a permanent record of all the transactions in a distributed, time-stamped and secure way over a peer-to peer network. Derived from various technologies, including cryptographic hash function, cryptographic digital signature, and distributed consensus, it affords some key functionalities such as data persistence, transparency, anonymity, integrity, and execution in a trustless environment. Moreover, the introduction of smart contracts has transformed the blockchain from a pure distributed database to an hybrid distributed storage and computing platform. The smart contracts are digital protocol that enforce agreements with terms and promise set by the involved parties. Such autonomous scripts ensure the execution of actions only after the verification of a specific set of conditions across the blockchain network. For such reasons, blockchain is an ideal candidate in the healthcare domain, in comparison with centralized or distributed databases integrated with access control mechanisms [7] due to the requirements of immutability and to the possibility to act, by means of the peer-to-peer infrastructure, across different stakeholders. Otherwise, access control systems usually refer to authorizations regulated by a single stakeholder, and make use of a data storage or are cloud-centric (which could represent a single point of failure in the network infrastructure) [8], but this is not the scenario analyzed in this paper.

In particular, the applications of blockchain technology in healthcare are categorized into supply chain management, patient data management, clinical trials and data security, drug traceability, claims adjudication, billing, and others. Supply chain management is considered the most promising application of blockchain technology in healthcare, as reported in [9]. Healthcare organizations face numerous challenges that come with new demands, such as customer dissatisfaction, rising healthcare costs, competition, and reduced reimbursement for services. All of these factors compel healthcare organizations to implement a system that can meet these demands and effectively deal with continuous changes, technological advancements, escalating healthcare costs, intense competition, and ensuring customer satisfaction. Healthcare organizations are increasingly turning to supply chain management as a mean of cost control and achieving their goals. Supply chain management encompasses the flows of goods, information, and funds between and within supply chain partners to meet consumer needs in the most efficient way possible. However, supply chain management in healthcare presents unique challenges due to the added risk and complexity, as a compromised supply chain can jeopardize patient safety as well as the tampering or violation of their medical information [9].

To cope with such issues, the solution proposed in this paper consists of the development of a medical record and supply chain management system based on the Hyperledger Fabric permissioned blockchain and Internet of Things (IoT) based architecture. Hence, the envisioned approach improves the level of security in terms of access control. In fact, access control policies, encapsulated into smart contracts, are defined to guarantee that only the users that have the correct permission can access the data, while transactions are stored inside the blockchain. The access control policies allow the definition of different roles, which are stored inside X.509 digital certificates. Such an IoT-driven blockchain approach aims to cope with the scalability and interoperability issues that emerge from the wide smart health scenarios. For such a reason, the consensus protocol will be developed to fulfill the needs of blockchain-oriented IoT-based healthcare applications. It can be conducted to gather consensus via data transaction validation rather than transaction syntax alone. Data acquisition from sensors and medical devices in the healthcare sector usually concerns a dense topology in certain areas (e.g., hospitals). Hence, the ability to construct data-centric consensus protocols is fundamental to interact with IoT technologies [10]. To assess the feasibility of the envisioned solution, the performance evaluation phase investigates the time and computational resources required to carry out the operations of creating, reading, and updating the assets, and reading the asset history, which are the tasks usually requested towards a blockchain. The scope of the approach proposed in this paper is to prevent possible attacks to data confidentiality and integrity, which are counteracted coupling the blockchain-based infrastructure with the access control management, responsible of cleverly regulating the resources' disclosure.

The remainder of this paper is organized as follows. Section 2 definitely clarifies the background and motivations behind this work, by analyzing the solutions that adopt blockchain in medical supply chains. Section 3 explains the choice of Hyperledger Fabric as the blockchain exploited in the proposed approach, along with the other involved technologies. Section 4 provides the details about the envisioned IoT-driven blockchain mechanisms and architecture for healthcare supply chain management; moreover, a discussion about the security features related to the presented work are hereby presented. Section 5 illustrates the obtained results. Finally, Section 6 end the paper and provides some hints for future research directions.

2. Background, related works and motivations

Healthcare applications serve various purposes, including decision-making, workflows, clinical data, electronic health records (EHR), genomics medicine, neuroscience, biomedical, and pharmaceuticals [11]. Standardizing data and communication protocols can help IoT technologies deliver efficient healthcare services. Improved connectivity, user interfaces, patient data security, and data interoperability can reduce the challenges of providing efficient healthcare services. Currently, healthcare is one of the most popular research domains, and researchers strive to create more reliable healthcare applications for the community and industry. Various stakeholders, such as patients, hospitals, and pharmacies, require secure maintenance, sharing, and access to health records without any alterations. Blockchain technology is emerging as a solution to address the challenges of the healthcare industry. The challenges of creating a system in a healthcare environment based on blockchain are [12]:

- **Security, privacy and anonymity of data**: Individual data must be privately used, and only authorized parties can access the requested data. The use of blockchain technology can eliminate the need for third-party intermediaries in transactions, but security and privacy issues remain a major challenge. While the decentralized nature of blockchain allows the community to identify blockchain structures, it also creates security and authentication risks. Without third-party administration, patients may need to choose multiple representatives to access their medical records in emergencies. This could lead to data security attacks if a group of people can access lists of similar patients. The limitations of blockchain technology lie in transmitting information to destinations with restricted data access. Moreover, for the sake of privacy, organizations should have no apparent identification. Absolute secrecy is difficult to achieve; hence, pseudo-anonymity is much more widespread [13]. To face this challenge, the solution envisioned in this paper includes the adoption of a consortium blockchain, in order to increase the trust among entities (which must be registered in order to interact with the network) and reduce the risk of interactions with malicious parties.
- **Authorization of the interacting entities and authenticity of the managed resources**: Before granting access to sensitive material, requestors' identification must be verified. Such a requirement is fundamental in order to avoid identity spoofing and impersonation attacks. Hence, in the proposed approach, it is mandatory for both users and stakeholders to register to the system, in order to start any interaction. Furthermore, also

the confidentiality and the integrity of the information managed inside the system must be protected. To this end, we integrate the use of digital certificates, which would be invalidated in case of tampering. In addition, cryptography also allows an entity to prove its identity, since other entities trust the certificate authority. Finally, access control policies included in smart contracts will ensure the correct regulation for data disclosure.

- **Compliance with public regulation**: The adoption of blockchain technology is in conflict with the General Data Protection Regulation (GDPR),[2] approved by the EU Parliament in 2016. In fact, the distributed nature of the blockchain does not comply with the requirement of entity's ownership for the managed data; moreover, the immutability of the stored transactions prevents the possibility to remove, upon a request, the information related to the requestor. To mitigate such an issue, this solution decides not to adopt a public blockchain, in order to limit the set of interacting parties. Instead, the privacy principles established by the Organisation for Economic Cooperation and Development (OECD)[3] are less restrictive with respect to such requirements, since they are focused on the specification of the purpose for data collection and the use limitation of information.
- **Managing the storage ability and the scalability of transactions**: The maintenance of storage capacity and speed of transaction processing are other issues in healthcare blockchain systems. Blockchain is designed to lead transaction information with restricted storage ability, but the healthcare sector has a large amount of information that requires significant storage space. Moreover, as the dataset increases, the speed of record discovery and accessing decreases, which is not appropriate for transactions that require high velocity. Therefore, maintaining storage capacity and transaction speed are critical issues that need to be addressed when implementing blockchain technology in healthcare. To cope with such as issues, several mechanisms can be adopted [14]. In our paper we focus on both on-chains and off-chains solutions. The former is related to reducing block data, while the latter consists in increasing the transaction throughput by executing the transaction outside of the main blockchain, by means of the distributed powerful smart node, described in Section 4.
- **Interoperability issues**: Blockchain technology also enables secure interoperability between healthcare organizations. Blockchain faced this problem by making blockchain from different connecting workers and applications to others at a faster rate. This issue generates interference in the efficient sharing of the information.
- **Auditability and accountability**: An individual or an institution should be investigated and held auditable and accountable for their actions.
- **Adjustment issues**: Blockchain technology is static in the beginning and applied in direction of the real development in medicine and healthcare which may certainly face adjustment problems. An amount of real authenticated and specialized standards may be required from worldwide standardization establishments. The already defined standards may be helpful to compute the dimension and set up the data interchanged in blockchain services. Such standards may not only examine the shared information, but also remain protective measures.
- **Social issues**: The adoption of blockchain technology in the medical industry faces both technological and social issues. It requires the establishment of specialized standards to compute the dimension and set up the data interchanged in blockchain services. Moreover, achieving technological approaches that are diverse from the standard research techniques may not be simple. The healthcare industry is moving slowly towards digital data, and there is still an issue with the complete adoption of blockchain technology in the medical sector due to the low acceptability value and the guidelines supplied. Encouraging medical experts to switch from research data to blockchain is a challenging task.

Focusing on the blockchain use case for EHR, several blockchain-based systems have been developed that aim to securely and efficiently manage medical data [11]. GEM is an example of a healthcare platform that enables decentralized management of medical records, data sharing, and authorization among different users of the system [15]. Healthbank is another blockchain-based healthcare platform that allows every patient to store and manage their medical data and provides healthcare data to researchers and pharmaceutical companies [15,16]. Another kind of platform is HDG (Healthcare Data Gateway) which is used to provide a secure and safe remote patient-monitoring system [17]. Personal Health Record (PHR) is another technology that allows patients to access and monitor their own health data, and also enables them to share such data with other healthcare professionals and personal contacts [18]. Blockchain technology is also emerging in the healthcare sector for the detection of medical fraud in the pharmaceutical industry because it is hard to detect fake drugs as these medicines came to the seller through a complex logistic system and it is impossible to track the authentic supply chain. The emergence of blockchain technology has revolutionized the traditional supply chain and can keep track of each party in the supply chain. MediLedger is an example of a blockchain-based system currently being implemented in the US pharmaceutical industry, comprising many industry leaders, that share and validate data across organizations [19]. Summarizing, existing architectures have emphasized the need for a robust EHR management and track and trace system for medical supply chains. An end-to-end managing system across the medical supply chain is paramount to ensuring patients' safety and eliminating counterfeits. Most existing and currently adopted EHR management and track and trace systems are centralized, leading to data privacy, transparency and authenticity issues in healthcare supply chains. The just presented platforms try to overcome this issue by adopting decentralized blockchain technology. However, they are targeted either on EHR management or pharmaceutical products' tracking and tracing. However, since patients are personally involved in drugs consumption, they should also be included throughout the system which control the provenance and delivery of drugs.

Moreover, concerning other solutions targeted to the management of medical supply chain through blockchain technology, the approach proposed in [20] adopts Ethereum platform to secure transactions for drug traceability. Ethereum blockchain is also exploited in [21], which presents an healthcare supply chain contracting process, involving multiple stakeholders such as manufacturers, distributors, and healthcare providers. The proposed solution guarantees that only registered stakeholders are allowed to register and interact with the smart contract, thus ensuring trust and transparency among stakeholders themselves. The authors of [22] also uses Ethereum platform and smart contract solutions to trace and track the information specifically related to blood donation supply chain. Ethereum and IoT are coupled together in [23] to obtain an IoT-driven blockchain-based efficient and reliable product tracking and tracing in the healthcare supply chain. The envisioned overall system is composed by healthcare products and devices equipped with sensors turned into smart connected devices and products, which have connectivity to IP network as well as to the Ethereum blockchain network. This would allow remote monitoring, control, and

[2] GDPR.EU https://gdpr.eu/.

[3] OECD Guidelines on the Protection of Privacy and Transborder Flows of Personal Data http://www.oecd.org/sti/ieconomy/oecdguidelinesontheprotectionofprivacyandtransborderflowsofpersonaldata.htm.

management of the healthcare products. An IoT computer hardware board with processing and networking capabilities will interface with the sensors on the product or device for reading and control, as well as the capability to communicate to the cloud and Ethereum blockchain. Such a solution is very promising, however it was only conceptualized, but neither prototyped or deployed.

Note that, in the healthcare supply chain, involving products and sensors that collect data with sensitive and private content, a permissioned blockchain networks such as *Hyperledger Fabric*, may be the better or more preferred approach. This is because only authorized and registered users and stakeholders have access to the ledger and its transactions as well as content. The work presented in [24] employs *Hyperledger Fabric* to secure and efficiently manage EHR systems; however, no implementation is provided. The Medledger framework, described in [25], focuses to solve the problems of drug traceability by storing and recording all drug-related activities, events, and transactions involving all the participating entities in pharmaceutical industry and supply chain.

Only some works couple blockchain and IoT in healthcare supply chain management. For example, the authors of [26] just investigate the IoT-based blockchain technologies for the pharmaceutical supply chain and its related issues. They provide extensive proposal of digital technologies, which would improve traceability and visibility as the pharmaceutical drug moved along the supply chain. The work, presented in [27], integrates Ethereum blockchain technology with IoT to achieve a decentralized tracking and tracing of the medical products, avoids counterfeit drugs, and provides the status of the products during the shipment process between manufacturers to end-user.

In light of existing approaches and technologies, the system developed in this work enables to manage both patient medical records and medical supply chain, exploiting an IoT-driven blockchain architecture. Upon registering with the system to a designated organization representing a hospital, a manufacturer, a distributor, or an healthcare providers, authorized users will be able to create, read, update, and access the medical records history, on the basis of their certified role. The system adopts *Hyperledger Fabric*, a consortium blockchain that facilitates access control management for sensitive records. As a result, only registered users can access the records. As just said, a permissionless blockchain is not employed in this case, as it would allow public access to the data, whereas the private nature of the medical records necessitates restricted access. The smart contract guarantees data provenance, eliminates the need for intermediaries and provides a secure, immutable history of transactions to all stakeholders, which are involved in the investigated scenario. The advantages related to the adoption of *Hyperledger Fabric* will be clarified in Section 3.

3. Background on involved technologies

Blockchain technology is a sophisticated database system that enables transparent information sharing among participants in a business network. The data is stored in blocks that are linked in a chain, forming a chronologically consistent ledger. This allows for the creation of an immutable record of transactions, such as orders, payments, and accounts. The consensus-based system provides protection against unauthorized transactions and ensures a shared, consistent view of the data for all network participants. The initial and widely known use of blockchain technology is the Bitcoin cryptocurrency.[4] However, other cryptocurrencies such as Ethereum[5] have also emerged, adopting similar features to Bitcoin, while incorporating smart contracts to establish a platform for decentralized applications. Bitcoin and Ethereum belong to the category of public permissionless blockchain. This means they are open and accessible to everyone and participants can interact without revealing their identities. As the popularity of Bitcoin and Ethereum grows, their enterprise's use cases grow with them. However, many enterprise use cases require performance characteristics that permissionless blockchain technologies are unable (presently) to deliver. For enterprise use, we need to consider the following requirements:

- Participants must be identified/identifiable.
- Networks need to be permissioned.
- High transaction throughput performance.
- Low latency of transaction confirmation.
- Privacy of transactions and data pertaining to business transactions.

In a permissionless blockchain, participation is open to almost anyone, and all participants remain anonymous [28]. Due to the lack of trust in this environment, the immutability of the blockchain's state before a certain point is the only certainty. To address this trust issue, permissionless blockchains often use a mined native cryptocurrency or transaction fees as an economic incentive to balance the high costs of participating in a consensus mechanism based on "proof of work" (PoW), that is tolerant to Byzantine faults. In contrast, permissioned blockchains run among a defined group of identified and often vetted participants, who are governed by a model that creates a certain level of trust [28]. These blockchains provide a secure way for a group of entities with a common goal to interact, even if they do not fully trust each other. Because the participants' identities are known, permissioned blockchains can use more conventional consensus protocols, such as crash fault-tolerant (CFT) or byzantine fault tolerant (BFT), that do not require expensive mining processes. Furthermore, the risk of a participant introducing malicious code through a smart contract is reduced in a permissioned environment. Since the participants are known to each other, all actions such as submitting transactions, changing the network configuration, or deploying a smart contract are recorded on the blockchain, following an endorsement policy established for the network and transaction type. Instead of being completely anonymous, the responsible party can be easily identified and the situation is dealt with according to the governance model's terms.

3.1. Hyperledger fabric

Hyperledger Fabric,[6] as a permissioned platform, provides confidentiality through its channel architecture and private data feature. Channels allow participants in a Fabric network to create a sub-network where each member has visibility to a specific set of transactions. This means that only nodes participating in a channel have access to the smart contract (i.e., the chaincode) and transacted data, preserving their privacy. Private data collections among members on a channel also provide similar protection as channels, but without the need for creating and maintaining separate channels, reducing the maintenance overhead. Unlike with a public permissionless network, the participants are known to each other, rather than anonymous and, therefore, fully untrusted. This means that while the participants may not fully trust one another (they may, for example, be competitors in the same pharmaceutical industry), a network can be operated under a governance model that is built off of what trust does exist between participants, such as a legal agreement or framework for handling disputes.

Hence, in this kind of blockchain the members are not unknown, but there is the presence of a trusted entity that decides who can enroll in the blockchain. Since only trusted entities can take part in the blockchain there is no need for protocols like PoW or proof of stake, in order to validate transactions and secure the network. Another distinctive feature is the channel, which consists of allowing a group of participants to create a separate ledger of transactions. This feature

[4] Bitcoin, https://developer.bitcoin.org/reference.

[5] Ethereum, https://ethereum.org/it/developers/docs.

[6] HyperledgerFabric, https://hyperledger-fabric.readthedocs.io/en/latest.

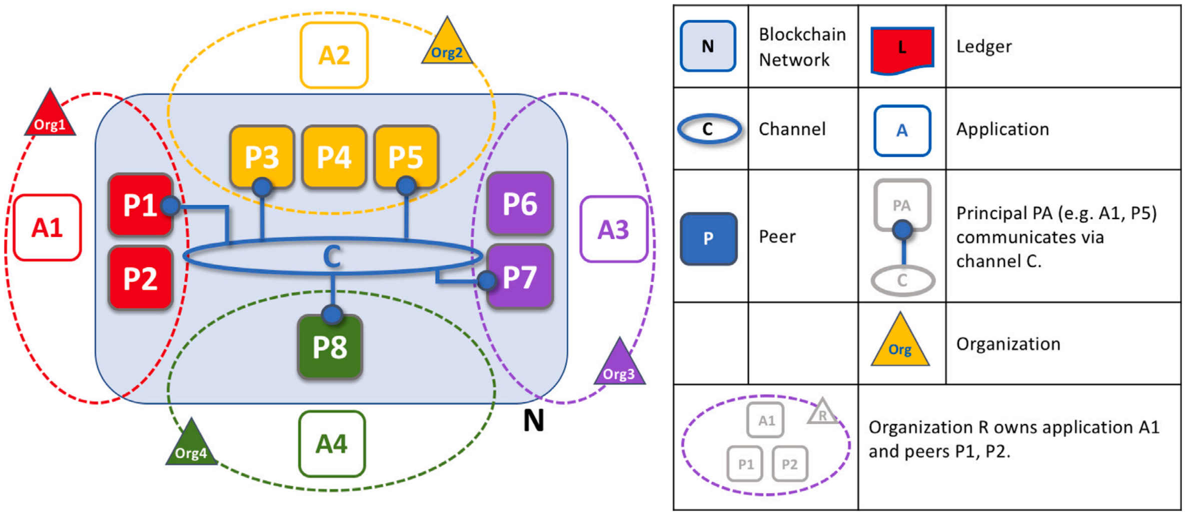


Fig. 1. Elements that constitute an *Hyperledger Fabric* network.

might be very useful for some networks where participants might be competitors and not want transactions they make known to every participant of the network. Fig. 1 shows an example of an *Hyperledger Fabric* network.

Hyperledger Fabric is also the first Distributed Ledger Technologies (DLT) platform to support smart contracts written in commonly used programming languages, such as *Java*, *Go*, and *Node.js*, eliminating the need for specialized training in new languages. This means that enterprises can leverage their existing skill sets to develop smart contracts, making the platform more accessible and efficient. Moreover, one of the most important of the platform's differentiators is its support for pluggable consensus protocols that enable the platform to be more effectively customized to fit particular use cases and trust models.

3.2. X.509 digital certificates

Every different active element inside or outside the *Hyperledger Fabric* network, that is able to consume services, has a digital identity, which is encapsulated in an X.509 digital certificate. Such digital identities are important because they determine the exact permissions over resources and access to information that actors have in a blockchain network. In order to determine these permissions the digital identity is made of attributes that *Hyperledger Fabric* uses to determine permissions. Attributes mainly consists in roles organization's membership of the entities acting inside the considered healthcare scenario (e.g., patients, medical staff, pharmacies, other manufactures, and so on).

In order to make a digital identity to be verifiable it must come from a trusted authority. This trusted authority in *Hyperledger Fabric* is called Membership Service Provider (MSP). More specifically, a MSP is a component that defines the rules governing the valid identities of an organization. The implementation of this trusted authority in *Hyperledger Fabric* uses X.509 certificates as identities and it adopts a traditional Public Key Infrastructure (PKI) hierarchical model. The elements that make up a PKI are the Certificate Authorities (CA), who issue digital certificates to parties to authenticate themselves in the messages they exchange into the network. A CA's Certificate Revocation List (CRL), constitutes a reference for the certificates that are no longer valid. Digital certificates are documents that hold a set of attributes relating to the holder of the certificate. The type of certificates used in *Hyperledger Fabric* is compliant with the X.509 standard. In a digital certificate all the information that is enclosed in the attributes is encrypted and doing this tampering will invalidate the certificate. In addition, cryptography allows an entity to prove its identity since other entities trust the certificate issuer (i.e., the CA). The last element that constitutes a PKI is the public and private keys pair. The authentication mechanism relies on digital signatures that require each entity to hold two cryptographically connected keys (i.e., private and public keys).

3.3. Hyperledger fabric network

Peers are a fundamental element of the network because they manage ledgers and chaincodes. More in detail, the peer hosts instances of the ledger and instances of chaincode: this is because blockchain technology requires consistent replicas of data and smart contracts. This design is useful in order to avoid single points of failure and to provide consistent ledgers. In addition, a peer is capable of hosting not only one ledger, but it has the ability to host multiple different ledgers since it can be part of different channels inside an *Hyperldger Fabric* network, as shown in Fig. 1. The same is for the chaincodes, in fact, a chaincode is instantiated on a single channel and a channel can have multiple chaincodes that interact with it and, consequently, a peer can host multiple chaincodes that communicate with the ledger. Moving to the end user side, a peer make the ledger accessible to the end user through applications that allow users to interact with the ledger. In order to create these applications there is a set of APIs in the *Fabric Software Development Kit*, that enable the applications to connect to peers, invoke chaincodes to generate transactions, and submit the transactions to the network that will get ordered, validated, and committed to the ledger. *Hyperledger Fabric* networks are administered by a collection of organizations and an organization can own one or more peers. Therefore, it is convenient to analyze the relationship between organizations and peers. More in detail, a peer is only an element of an organization and, as a consequence, the peer can be considered as one of the resources that constitute an organization. The other resources owned by an organization are: (i) the *World State Database*, which is a database that owns the last state of an asset present in the ledger; (ii) the CA, which is responsible to distribute digital certificates; (iii) and the MSP, whose aim is to assign roles inside the network. Whenever a peer connects to a blockchain network, a policy in the channel configuration uses the peer's identity to determine its rights. This identity is provided to the peer by the CA, which gives to the peer a digital certificate. The mapping of identity to the organization is provided by the MSP, which determines how a peer gets assigned to a specific role in a particular organization and, accordingly, gains appropriate access to blockchain resources.

Another essential element is the ledger, because it stores important information about business objects, in particular, it contains the current state of the objects as a journal of transactions. In the case of

Hyperledger Fabric, a ledger consists of two distinct elements: the world state and the blockchain. The first is a database that holds the current values of a set of ledger states and it is useful because it makes easy access to the current value of a state which would otherwise have to be calculated by traversing the entire transaction log. The second element, the blockchain, is a transaction log recording all the changes that have resulted in the current world state. The blockchain data structure is very different to the world state because it is immutable.

3.4. Smart contracts, chaincode and ordering service

Smart contracts and chaincode are interchangeable terms in *Hyperledger Fabric*, but in general smart contracts define the executable logic that generates new transactions that are added to the ledger; this means that they control the lifecycle of a business object contained in the world state database. On the other hand, the chaincode is used to group related smart contracts in order to be deployed to a blockchain network. More in detail, a smart contract programmatically accesses two distinct pieces of the ledger: the blockchain and the world state. In particular, it can access the world state by doing a query in order to retrieve information about the current state of a business object, and it can access the blockchain by creating transactions. Once smart contracts are written and are grouped in a chaincode, the chaincode is then associated with an endorsement policy that applies to all the smart contracts defined within it. The endorsement policy is so important because it indicates which organizations in a blockchain network must sign a transaction generated by a given smart contract in order to declare valid the transaction itself.

Smart contracts are the main part of the development of an application that runs on *Hyperledger Fabric*, since they contain a set of transaction definitions, and deploying a chaincode to a network makes effective its smart contracts to the organization in that network. Associated with every chaincode, as already said, there is an endorsement policy that is applied to all of the smart contracts defined within it. An endorsement policy indicates which organizations in a blockchain network must sign a transaction generated by a given smart contract, in order for that transaction to be declared valid. By analyzing the smart contract execution, it is relevant to say that the smart contract runs on a peer node inside the blockchain network. The contract takes a set of input parameters, that are called transaction proposals, and are used in order to read and write the ledger. Every change to the world state database is captured as a transaction response, that contains a read-and-write set with both the states that have been read and the new states that are to be written if the transaction is valid.

It is noticeable to say that the world state database is updated only after the validation of the transaction and not when the smart contract is executed. The validation of a transaction consists of two phases. In the first phase, the transaction is distributed to all peer nodes in the network and is checked to ensure it has been signed by sufficient organizations according to the endorsement policy. In the second phase, the transaction is checked to ensure that the current value of the world state database matches the read set of the transaction, when it is signed by the peer node. After these two phases, the transaction is added to the blockchain history, whether it is valid or invalid, but only valid transactions result in an update to the world state database.

The last element in *Hyperledger Fabric* platform is the ordering service. As already mentioned *Hyperledger Fabric* blockchain network is permissioned and this means that the blockchain does not rely on probabilistic consensus algorithms, like *Ethereum* and *Bitcoin*, which guarantee ledger consistency to a high degree of probability, but this makes them vulnerable to divergent ledgers, also called forks. In this sense *Hyperledger Fabric* is different, in fact, it has a particular node, called $orderer$, which deals with the ordering of the transactions, and along with the other orderer nodes form an ordering service. Another difference between *Hyperledger Fabric* and other permissionless blockhain systems is the consensus algorithm, which is deterministic and not probabilistic. The use of a deterministic consensus algorithm ensures that any block validated by the peer is guaranteed to be correct and this prevents the formation of ledger forks. As just mentioned, when a transaction proposal is sent by the client to a trusted peer in the network, this transaction is forwarded to the other endorsement peers that then answer with the transaction response. After this process, the transaction is sent to the ordering service, which orders it with other endorsed transactions and packages them all into a block. This block is then saved to the orderer's ledger and distributed to all the peers in the channel. It is important to notice that the ordering service puts the transactions into a strict order and peers use this order when validating and committing transactions. The latest version of *Hyperledger Fabric* uses the Raft protocol [29] to order transactions. Raft follows a "leader and follower" model, where a leader node is elected, for each channel, and its decisions are replicated by the followers.

3.5. Node.js

Node.js[7] is a framework for building web applications in *JavaScript*, allowing us to use this language, typically used client-side, also for writing server-side applications, through the use of the *JavaScript Engine V8*. The main feature of *Node.js* consists in the possibility it offers of accessing operating system resources in event-driven mode and not by exploiting the classic model based on concurrent processes or threads. Each action is therefore asynchronous, and this should ensure a certain efficiency of applications due to a callback system managed at a low level by the runtime. In order to manage the asynchronous operations *Node.js* uses the *Event Loop*. When asynchronous operations terminate *Node.js* is notified via events. Such events will be placed inside a queue and the respective callback functions, recorded when writing the program, will be executed one at a time until the call stack of the *Node.js V8 Engine* is empty.

3.6. Go programming language

Go[8] is a procedural, statically typed, concurrent, and garbage-collected programming language. It is designed to be simple, efficient, and easy to learn, making it a popular choice for building scalable network services, web applications, and command-line tools. *Go* is known for its support for concurrency, which is the ability to run multiple tasks simultaneously. Concurrency is achieved in *Go* through the use of Goroutines and channels, which allow the developer to write code that can run multiple operations at the same time. This makes *Go* an ideal choice for building high-performance and scalable network services, as well as for solving complex computational problems. Another important feature of *Go* is its garbage collection, which automatically manages memory for the developer. This eliminates the need for manual memory management, reducing the likelihood of memory leaks and other bugs that can arise from manual memory management.

3.7. MongoDB

MongoDB[9] is an open-source *NoSQL* database. As a non-relational database, it is able to process structured, semi-structured, and unstructured data. It uses a non-relational, document-oriented data model and an unstructured query language. Documents consist of key–value pairs, which are the basic unit of data in MongoDB. Each document is stored in JSON format and is basically a tree that can contain a lot of data. Documents are grouped in collections that can also be heterogeneous. This means that there is no fixed schema for documents. There are no relationships or bindings between collections guaranteed by *MongoDB*. The key features of MongoDB are:

[7] Node.js, https://nodejs.org/en.

[8] Go programming language, https://go.dev.

[9] MongoDB, https://www.mongodb.com/.

- High availability services, since replication of a database can be done very easily
- Automatic scalability, which is the ability to distribute collections in clusters of nodes, so as to support large amounts of data without heavily affecting performance.

4. Proposed solution

In this section, the application scenario, the blockchain implementation, along with the smart contract definition, and the envisioned IoT-driven blockchain architecture are described.

4.1. Architecture

Fig. 2 represents the structure of the high-level system, and how the developed entities interact, while the sequence diagram, sketched in Fig. 3, shows the overall workflow and interactions among the system's components. Note that the system includes "edge" end-IoT devices, which are responsible for sensing and acquire data from people or from the environment where they are placed in, and "powerful" IoT devices, that constitute the network running the blockchain framework, as detailed in this Section. Moreover, the actions towards the blockchain can be triggered by end-IoT devices which are connected both to users (e.g., patients or medical staff) and to stakeholders (e.g., medical service providers, laboratories, pharmacies, research centers, insurance).

The entities shown in Fig. 2 have been presented in Section 3. Note that, a web application is provided to allow an easy interaction of users with the healthcare management system. As shown in Figs. 7 and 8, the graphical user interface is simple and intuitive to understand, and allows to gather the information of interest in few clicks. A future development surely will be the improvement of the quality of the layout as well as the development of a related mobile app. Tips and textual suggestion will be added in order to facilitate the utilization to tech-savvy healthcare workers. Users' sessions and information are stored in a MongoDB storage. One or more organizations, and the related variable number of peers, can take part to the network.

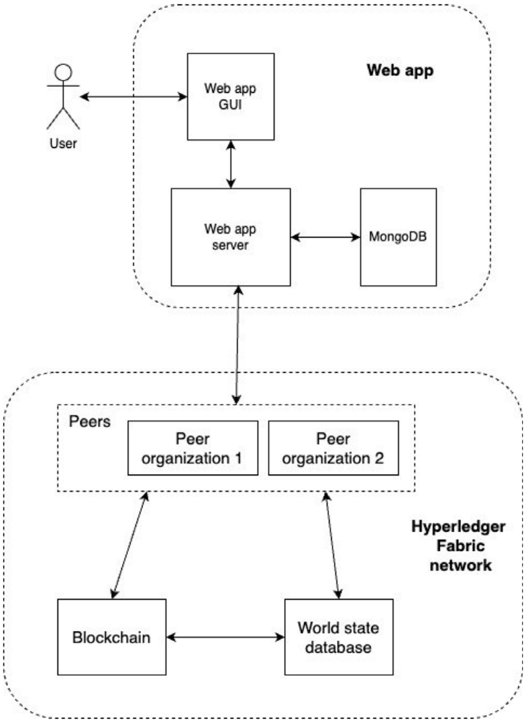


Fig. 2. High-level scheme of the architecture.

4.2. Application scenario

The application scenario is placed in the healthcare sector, where the management of medical records and supply chain is allowed to patients, medical staff and stakeholders, registered to different medical centers, manufacturers, distributors, and healthcare providers. Such users can perform the operations of creation, reading, and updating of medical records and products data, while administrators can, in addition, consult all the modifications and operations that have been made on a given record or data through the read asset history operation. Hence, the former type of users will access the world state database content, while the latter will access the blockchain's transactions history. The network will manage both medical records of patients and drugs supply chain, even in the same or in separate organizations or channels, inside the same organizations, but with different associated peers.

Medical records have the following sample structure (fields can be changed/added/removed):

- **Tax Number**: "Tax Number" is the patient identifier and it is represented by a string.
- **Smoking**: "Smoking" defines whether the patient is a smoker. The data is represented by a string "Yes" or "No".
- **Alcohol drinking**: "Alcohol drinking" defines whether the patient habitually drinks alcohol or not. The data is represented by a string "Yes" or "No".
- **Walking difficulty**: "Walking difficulty" defines if the patient has difficulty walking. The data is represented by a string "Yes" or "No".
- **Physical activity**: "Physical activity" defines whether the patient habitually performs the physical activity. The data is represented by a string "Yes" or "No".
- **Asthma**: "Asthma" defines whether the patient suffers from asthma. The data is represented by a string "Yes" or "No".
- **Disease**: "Disease" defines one or more disease affecting the patient. The data is represented by an array of type string.
- **Therapy**: "Therapy" defines one or more therapies assigned to the patient. The data is represented by an array of type string.
- **Drugs**: "Drugs" defines one or more drugs assumed by the patient. The data is represented by an array of type string.
- **Vital signs**: "Vital signs" includes aggregated information (e.g., temperature, pressure) about the status of the patient. Such data could be transmitted periodically by IoT devices connected to the patient's himself/herself. The data is represented by an array of type string.
- **Medical reference**: "Medical reference" represents the medical reference center of the patient and it is represented by a string.

Instead, records related to products managed inside the medical supply chain have the following sample structure (fields can be changed/added/removed as well):

- **Serial Number**: "Serial Number" is the product identifier and it is represented by a string.
- **Batch Number**: "Batch Number" is the identifier of a quantity of items belonging to a single group.
- **Lot Number**: "Lot Number" is an identification number assigned to a particular quantity, batch or lot of a product from a single manufacturer.
- **Name**: "Name" is a string representing the name of the product.
- **Price**: "Price" is a double representing the price of the product.
- **Expiration Date**: "Expiration Date" specifies the expected product's expiry date.
- **Producer**: "Producer" is a string representing the name of the producer.

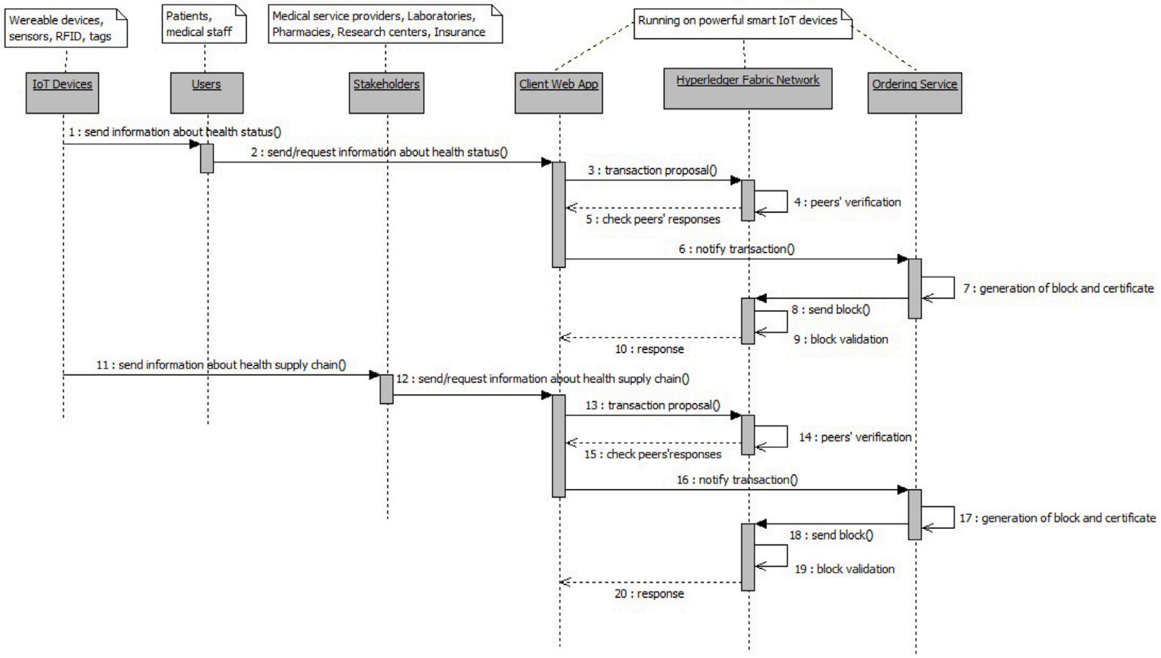

Wereable devices, sensors, RFID, tags
Patients, medical staff
Medical service providers, Laboratories, Pharmacies, Research centers, Insurance
Running on powerful smart IoT devices
IoT Devices
Users
Stakeholders
Client Web App
Hyperledger Fabric Network
Ordering Service
1 : send information about health status()
2 : send/request information about health status()
3 : transaction proposal()
4 : peers' verification
5 : check peers' responses
6 : notify transaction()
7 : generation of block and certificate
8 : send block()
9 : block validation
10 : response
11 : send information about health supply chain()
12 : send/request information about health supply chain()
13 : transaction proposal()
14 : peers' verification
15 : check peers'responses
16 : notify transaction()
17 : generation of block and certificate
18 : send block()
19 : block validation
20 : response

Fig. 3. Sequence diagram representing the overall workflow and interactions among the system's components.

4.3. Blockchain and chaincodes implementation

As explained in Section 3, the *Hyperledger Fabric* network consists of some different elements that are: the ledger, the ordering service, the certificate authority, and the peers. The network has two or more peers and one or more orderer that helps to run and maintain the network by verifying and endorsing the transactions. Smart contracts, written in *Go* language, are made with all the functionality needed to read and write the ledger. Fig. 4 shows how all the elements described above interact with each other in order to validate a transaction:

(1) The client (i.e., any entity interacting within the considered scenario) makes a transaction proposal. This proposal is signed with the user's certificate and sent to the peers.
(2) Each peer verifies the identity and the authorization of the user from the proposal. If all verification checks are successful, the peer generates a response using its certificate.
(3) The client collects and checks the peers' responses.
(4) The client sends the transaction response to the ordering service.
(5) The ordering service orders the generation of a new block of the received transactions and signs the generated block with its certificate.
(6) The ordering service broadcasts the generated block to all the peers.
(7) Each peer compares each transaction with its ledger world state. If the verification check is successful, the transaction is marked as valid and the world state of each peer is updated. Otherwise, the transaction is marked as invalid without updating the world state. Finally, the received block is appended to each peer's local blockchain.
(8) Finally, the client receives a response.

An important part of the work is the creation of access policies that allow the blockchain to verify that the user who is performing an operation can actually do it. Hence, after the creation of the network, a chaincode is defined in *Go* language, in order to interact with the blockchain. The aim of the chaincode is to manage the records that are stored inside the blockchain. More in detail, in order to implement the smart contract, the package "fabric-contract-api-go"[10] is used. It is the *Fabric Gateway client API* that allows the development of applications in *Go* language. This API uses the gateway peer capability, introduced in Fabric v2.4, to interact with the Fabric network. The records that are stored in the blockchain are represented by a struct in *Go* language. The listing of code 1 describes how a patient's record is represented:

```
type Asset struct {
  Action     string ‘json:"Action"‘
  User     string ‘json:"User"‘
  Organization     string ‘json:"Org"‘
  MedicalRecord     MedicalRecord
}

type MedicalRecord struct {
  TaxNumber     string ‘json:"TaxNumber"‘
  Smoking     string ‘json:"Smoking"‘
  AlcoholDrink     string ‘json:"AlcoholDrink"‘
  DiffWalking     string ‘json:"DiffWalking"‘
  PhysicalAct  string ‘json:"PhysicalAct"‘
  Asthma     string ‘json:"Asthma"‘
  Disease     [string] ‘json:"Disease"‘
  Therapy     [string] ‘json:"Therapy"‘
  Drugs     [string] ‘json:"Drugs"‘
  MedicalRef     string ‘json:"MedicalRef"‘
  VitalSigns     [string] ‘json:"VitalSigns"‘
}
```

Listing 1: Medical record structure

Instead, the listing of code 2 describes how a product's record is represented:

[10] Fabric Go API, https://pkg.go.dev/github.com/hyperledger/fabric-contract-api-go/contractapi.

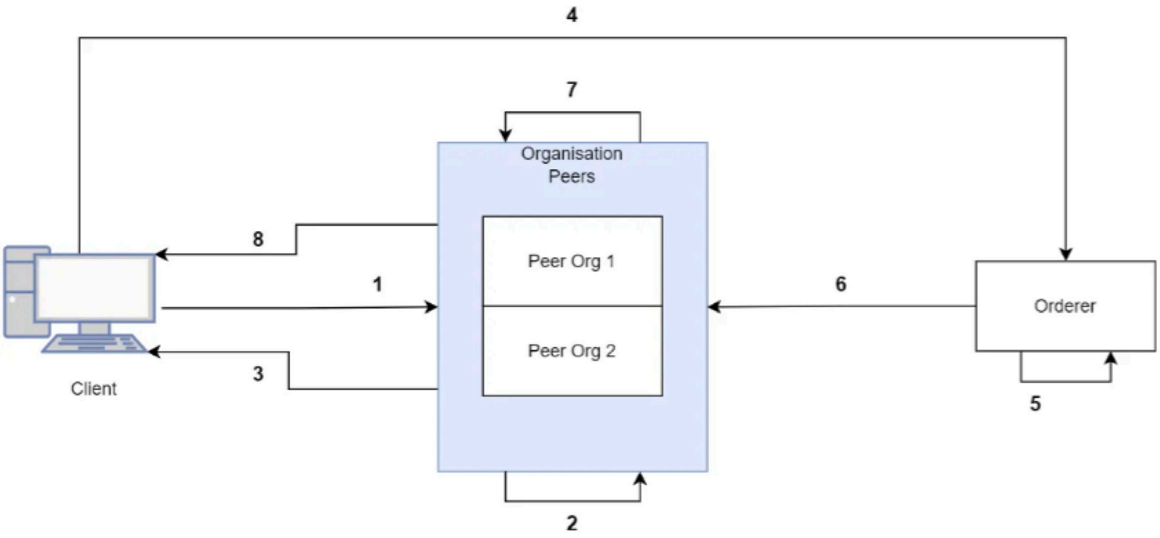


Fig. 4. Consensus schema.

```
type Asset struct {
  Action    string `json:"Action"`
  User    string `json:"User"`
  Organization    string `json:"Org"`
  ProductRecord    ProductRecord
}

type ProductRecord struct {
  SerialNumber    string `json:"SerialNumber"`
  BarchNumber    int `json:"BarchNumber"`
  BarchNumber    int `json:"BarchNumber"`
  Name    string `json:"Name"`
  Price    double `json:"Price"`
  ExpirationDate timestamp `json:"ExpirationDate"`
  Producer    string `json:"Producer"`
}
```

Listing 2: Product record structure

The fields that wrap records are used in order to track which kind of action is made, to track who made the action and to track which organization the record belongs to. The action that a user can do on a record is the creation, update, and read. The listing of code 3 represents the function in the smart contract, used to create/update an asset related to a medical record (the code for the product record is similar). Note that assets are the representations of values that are generated and stored digitally into the blockchain. It is worth to remark that, in order to store the data in the blockchain, the method *PutState* has been defined in the class *TransactionContextInterface*.

```
func (s *SmartContract) CreateAsset (
  ctx contractapi.TransactionContextInterface,
  user string,
  taxNumber string,
  smoking string,
  alcoholDrinking string,
  diffWalking string,
  physicalActivity string,
  asthma string,
  disease  [string],
  therapy  [string],
  drugs  [string],
  medicalRef  string,
  vitalSigns  [string],
) error {
  exists, err := s.AssetExists(ctx, taxNumber)
  if err != nil {
     return err
  }
  if exists {
     return fmt.Errorf("asset %s already exists",
                                       taxNumber)
  }
}
  MSPID, _ := ctx.GetClientIdentity().GetMSPID()

  asset := Asset {
  Action:  "CREATE",         //or "UPDATE"
  User:  user,
  Organization:  MSPID,
  MedicalRecord:  MedicalRecord{
  TaxNumber:  taxNumber,
  Smoking:  smoking,
  AlcoholDrinking:  alcoholDrinking,
  DiffWalking:  diffWalking,
  PhysicalActivity:  physicalActivity,
  Asthma:  asthma,
  Disease:  disease,
  Therapy:  therapy,
  Drugs:  drugs,
  MedicalRef:  medicalRef,
  VitalSigns:  vitalSigns,
  },

  assetJSON, err := json.Marshal(asset)
  if err != nil {
     fmt.Println(err)
     return err
  }
  return ctx.GetStub()
   .PutState(asset.MedicalRecord.TaxNumber,
                                 assetJSON)
```

Listing 3: Smart contract for the create/update function

Inside the method *AssetExists*, there is the implementation of an access control policy, which consists of checking if the "Organization" field of the asset matches the ID of the MPS of the user. Such a function is shown in the listing of code 4 (the code for the product record is similar).

```
func (s *SmartContract) AssetExists (
  ctx contractapi.TransactionContextInterface,
  taxNumber string,
) (bool, error) {
  assetJSON, err := ctx.GetStub().
                             GetState(taxNumber)
  if err != nil {
     return false,
     fmt.Errorf("failed to read from world state")
  }
  if assetJSON == nil {
    return false, nil
  }
  var asset Asset
  err = json.Unmarshal(assetJSON, &asset)
  if err != nil {
     return true, err
  }

  MSPID, _ := ctx.GetClientIdentity().GetMSPID()
  if asset.Organization != MSPID {
    return true, &AccessDenied{}
  }
  return true, nil
}
```

Listing 4: Function to check if a record exists

The other method used to access the blockchain is the read, which allows a user to obtain information on a record. This function is represented in the listing of code 5 for medical records (the code for the product record is similar). The code is similar for products' records, except for the research, which is performed using the product's serial number.

```
func (s *SmartContract) ReadAsset (
  ctx contractapi.TransactionContextInterface,
  user string,
  taxNumber string,
) (*MedicalRecord, error) {
  assetJSON, err := ctx.GetStub().
                             GetState(taxNumber)
  if err != nil {
    return nil,
    fmt.Errorf("failed to read from world state")
  }
  if assetJSON == nil {
    return nil,
    fmt.Errorf("the asset %s does not exist",
                taxNumber)
  }

  var asset Asset
  err = json.Unmarshal(assetJSON, &asset)
  if err != nil {
    return nil, err
  }

  MSPID, _ := ctx.GetClientIdentity().GetMSPID()
  if asset.Department != MSPID {
    return nil, &AccessDenied{}
  }

  asset.Action = "READ"
  asset.User = user
  assetJSON, err = json.Marshal(asset)
  if err != nil {
    return nil, err
  }

  err = ctx.GetStub()
   .PutState(asset.MedicalRecord.TaxNumber,
               assetJSON)
  if err != nil {
    return nil, err
  }
  return &asset.MedicalRecord, nil
}
```

Listing 5: Smart contract for the read function

Also in this method, there is an access control policy in order to check if the user that is making the read operation has the correct accesses. It is relevant to clarify that, for each of these methods, there is a corresponding transaction that tracks which user performed an operation on a specific record. In order to retrieve all these actions, that can be done on the existing records, another method is implemented, that is called $GetAssetHistory$ and is represented in the listing of code 6 (the code for the product record is similar). Note that, as just said, only the role "Administrator" for each organization/channel can access the assets' history. When a user is created, in fact, a role is associated, on the basis of his/her functions in the considered scenario (e.g., patient, member of the medical staff, and so on), as clarified later.

```
func (s *SmartContract) GetAssetHistory (
  ctx contractapi.TransactionContextInterface,
  taxNumber string,
) ([]HistoryQueryResult, error) {
  position, _, _ := ctx.GetClientIdentity()
                   .GetAttributeValue("Role")
  if position != "Administrator" {
    return nil, &AccessDenied{}
  }

  resultsIterator, err := ctx.GetStub().
                   GetHistoryForKey(taxNumber)
  if err != nil {
    return nil, err
  }
  defer resultsIterator.Close()

  var records []HistoryQueryResult
  MSPID, _ := ctx.GetClientIdentity().GetMSPID()
  for resultsIterator.HasNext() {
    response, err := resultsIterator.Next()
      if err != nil {
        return nil, err
      }
    var asset Asset
    var timestamp time.Time
    if len(response.Value) > 0 {
      err = json.Unmarshal(response.Value, &asset)
      if err != nil {
        return nil, err
      }
     if asset.Department != MSPID {
        return nil, &AccessDenied{}
     }
     timestamp = response.Timestamp.AsTime()
    } else {
      asset = Asset{
        Action: "HISTORY",
      }
      timestamp = time.Now()
      }

      record := HistoryQueryResult {
        TxId:       response.TxId,
        Timestamp:  timestamp,
        Record:     &asset,
        IsDelete:   response.IsDelete,
      }
      records = append(records, record)
      }
    return records, nil
}
```

Listing 6: Smart contract for medical record history retrieval

By deploying the chaincode containing all the described smart contracts, all the functions described above are available to the users that have access to the network, in order to interact with the blockchain.

4.4. Web app

The Fabric Gateway[11] API is used in order to interact with the peers of the network by calling the smart contracts. The Fabric Gateway is a service, introduced in *Hyperledger Fabric* v2.4, that provides a simplified, minimal API for submitting transactions to a Fabric network. More in detail, these APIs allow the client application to endorse a transaction proposal, submit a transaction and wait for the commit status event. All of these actions are defined in the method of the Fabric Gateway called $SubmitTransaction$, as shown in the listing of code 7.

```
const result = await contract.submitTransaction(
  "ReadAsset",
  `${user}`,
  `${taxNumber}`,
);
console.log(`Transaction has been evaluated,
   result is: ${result.toString()}`);
```

Listing 7: Function related to transaction's submission

Such a code represents the invocation of the $ReadAsset$ method defined in the chiancode, but this API is used also by the $CreateAsset$ and $UpdateAsset$ methods, since each of them creates a transaction that is stored inside the blockchain, in order to track every change that is made to records. In addition, the web application allows the creation of users. It is realized with Node.js framework and, as a database where user sessions and user information are stored, MongoDB is adopted. In order to create users inside the *Hyperledger Fabric* network, the web app uses the $FabricCAServices$ library, which is an implementation of the member service client which communicates with the Fabric CA server. Therefore, by using this client, a user can be associated with his/her own certificate and role, and, with the certificate and the role, a user can interact with peers and invoke the smart contracts.

In the following, first, it is explained how the digital certificate management works in *Hyperledger Fabric*. Then it is described how such certificates are used in order to implement access control policies.

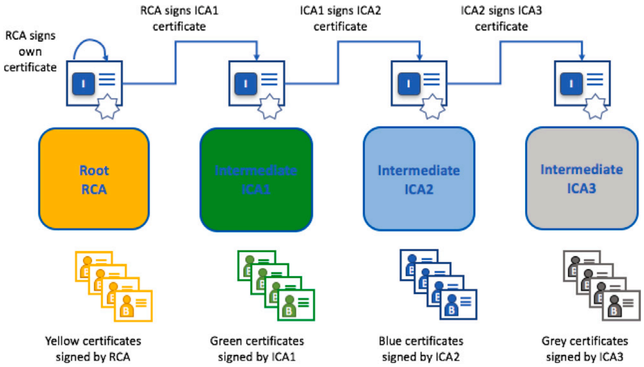

RCA signs own certificate
RCA signs ICA1 certificate
ICA1 signs ICA2 certificate
ICA2 signs ICA3 certificate
Root RCA
Intermediate ICA1
Intermediate ICA2
Intermediate ICA3
Yellow certificates signed by RCA
Green certificates signed by ICA1
Blue certificates signed by ICA2
Grey certificates signed by ICA3

Fig. 5. Elements that constitute a chain of Certificate Authorities.

4.5. Certificates' management in hyperledger fabric

There are different entities inside an *Hyperledger Fabric* network (i.e., peers, orderers, client applications and administrators). Each actor represents an active element inside or outside the network, able to consume services, and he/she has a digital identity encapsulated in an X.509 digital certificate. Such identities determine the exact permissions over resources and access to information stored into the blockchain network. A digital identity has some additional attributes that *Hyperledger Fabric* uses to determine permissions: the organization to which the user refers, and the role inside the organization. In addition, an identity, in order to be verifiable, must come from a trusted authority. This trusted authority in *Hyperledger Fabric* is the MSP, as described in Section 3.

More in detail, the X.509 certificates are issued by a CA. This means that in *Hyperledger Fabric* a digital identity has the form of cryptographically validated digital certificates that comply with X.509 standard and the MSP is in charge of establishing which identity is valid and which is not valid. In particular, CA issues identities by generating a public and private key which forms a key pair that can be used to prove identity. Such an identity needs a way to be recognized by the network, which is where the MSP comes in. For example, a peer uses its private key to digitally sign, or endorse, a transaction. The MSP is used to check that the peer is allowed to endorse the transaction. The public key from the certificate of the peer is then used to verify that the signature attached to the transaction is valid. Thus, the MSP triggers the mechanism that allows that identity to be trusted and recognized by the rest of the network. In the setting presented in this Section, one or more CAs can be used to define the members of an organization from a digital perspective. In particular, a CA can be put in place for each different organization, since operations are available inside the same organization to which a user belongs to.

Considering the case of internet users, the CAs came in two forms: the Root CA and the Intermediate CA. Since Root CAs have to securely distribute hundreds of millions of certificates, it makes sense to spread this process out across Intermediate CAs. Such Intermediate CAs have their certificates issued by the root CA or another intermediate authority, allowing the establishment of a chain of trust for any certificate that is issued by any CA in the chain, as shown in Fig. 5. This ability to track back to the Root Certificate Authority not only allows the function of Certificate Authorities to scale while still providing security because it limits the exposure of the Root Certificate Authority, which, if compromised, would endanger the entire chain of trust. If an Intermediate Certificate Authority is compromised, on the other hand, there will be a much smaller exposure.

Hyperledger Fabric provides a built-in CA component to allow the developer to create CAs in the blockchain networks. This component, called Fabric CA, is a private Root CA provider, capable of managing the digital identities of *Hyperledger Fabric* users that have the form of

[11] Fabric Gateway, https://hyperledger-fabric.readthedocs.io/en/latest/gateway.html.

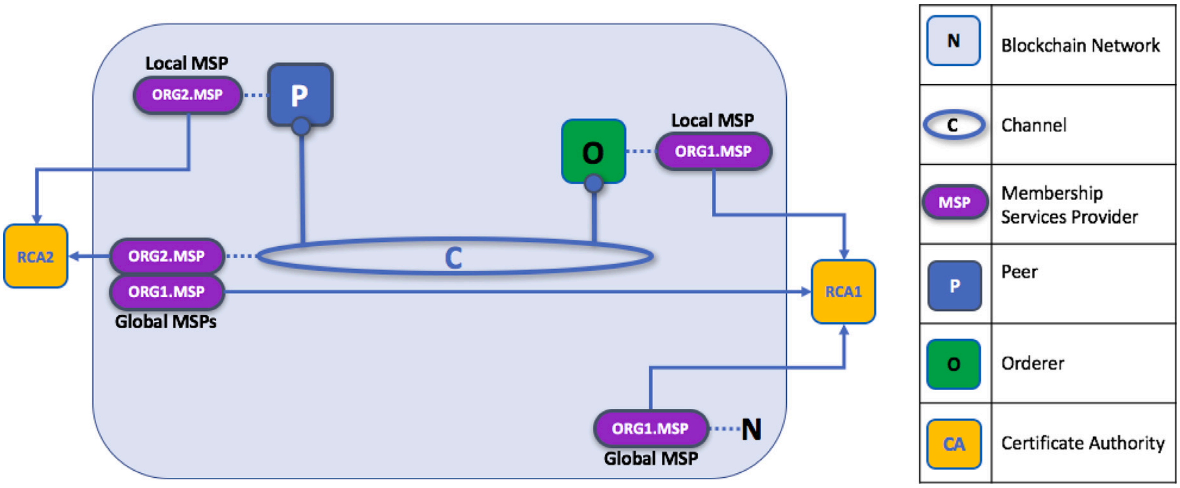


Fig. 6. Local and channel Membership Service Providers inside the Hyperledger Fabric network.

X.509 certificates. Whereas CAs generate the certificates that represent identities, the MSP contains the list of permissioned identities. The MSP identifies which Root CAs and Intermediate CAs are accepted to define the members of a trusted domain by listing the identities, or by identifying which CAs are authorized to issue valid identities for their members. However, the power of the MSP goes beyond simply listing who is a network participant or member of a channel. In fact, the MSP turns an identity into a role by identifying specific privileges an actor has on a channel.

Inside an *Hyperledger Fabric* blockchain network, the MSP is present in two different kind of domains: the local MSP and the channel MSP. Local MSP allows a user to authenticate himself/herself in his/her transactions as a member of a channel or as the owner of a specific role in the organization. Channel MSP defines administrative and participatory rights at the channel level. Peers and ordering nodes on an application channel share the same view of the channel MSP, and will therefore be able to correctly authenticate the channel participants. This means that, if an organization wishes to join the channel, a MSP incorporating the chain of trust for the members of the organization would need to be included in the channel configuration. Otherwise, transactions originating from this organization's identities will be rejected. Whereas the local MSP is represented as a folder structure on the file system, the channel MSP is described in a channel configuration. Therefore, the channel MSP defines the relationship between the identities of channel members, which themselves are MSPs, and the enforcement of channel-level policies. Channel MSPs contain the MSPs of the organizations of the channel members. Every organization participating in a channel must have a MSP defined for it. In fact, it is recommended that there is a one-to-one mapping between organizations and MSPs. The MSP defines which members are empowered to act on behalf of the organization. The channel MSP includes the MSPs of all the organizations on a channel. This includes not just the peers of the organizations, which are used to invoke chaincodes, but also the ordering service. Fig. 6 shows how local and channel MSPs coexist inside the *Hyperledger Fabric* network.

Note that an organization is a logically managed group of members. Hence, organizations manage their members under a single MSP. The MSP allows an identity to be linked to an organization. As a consequence, all of the identities that are part of a MSP, as identified by the Root CA and Intermediate CA folders, will be considered members of the organization.

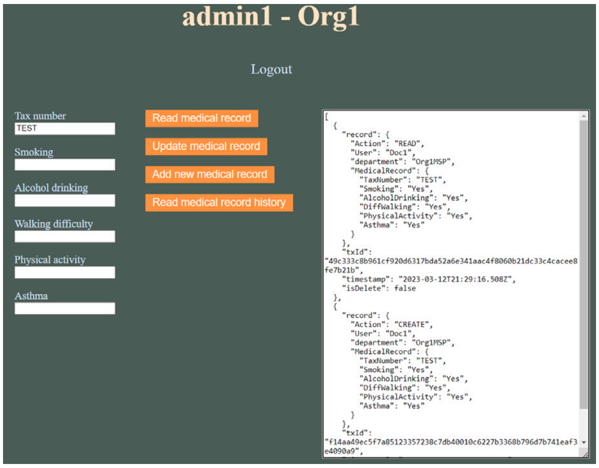
4.6. Access control policies implementation

As just described, in the certificates, issued by the CA, there is the possibility to specify some attributes that are associated with the user. In the X.509 certificates there the possibility to specifies four different roles: client, peer, admin, and orderer. Such roles are predefined inside the network and correspond to the active elements of the network. Users associated with the role "client" call the smart contracts and the following listing of code 8 shows how a user digital certificate is created, along with the role and organization assignment. Note that the role of an entity inside the *Hyperledger Fabric* blockchain network is different from the concept of the role, which is related to the healthcare application scenario (e.g., patient, medical staff, and so on), as shown in the listing of code 8.

```
let registerAttrs = [];
  let registerAttribute = {
    name: "Organization",
    value: "Patient",
    ecert: true
  };
  registerAttrs.push(registerAttribute);
  const secret = await ca.register({
  affiliation: ‘${this.orgId.toLowerCase()}.
                    organization1‘,
   enrollmentID: username,
   role: "client",
   attrs: registerAttrs
  }, adminUser);
  const enrollment = await ca.enroll({
    enrollmentID: username,
    enrollmentSecret: secret
  });
  const x509Identity = {
     credentials: {
         certificate: enrollment.certificate,
         privateKey: enrollment.key.toBytes(),
     },
     mspId: ‘${this.orgId}MSP‘,
     type: ’X.509’,
     };
   await wallet.put(username, x509Identity);
```

Listing 8: Creation of a user

Digital certificates are stored in a folder called *wallet* that then can be used to retrieve the certificates when they are needed. *Hyperledger Fabric* also gives the possibility to store such certificates in a separate



```
[
  {
    "record": {
      "Action": "READ",
      "User": "Doc1",
      "department": "Org1MSP",
      "MedicalRecord": {
        "TaxNumber": "TEST",
        "Smoking": "Yes",
        "AlcoholDrinking": "Yes",
        "DiffWalking": "Yes",
        "PhysicalActivity": "Yes",
        "Asthma": "Yes"
      }
    },
    "txId":
"49c333c8b961cf920d6317bda52a6e341aac4f8060b21dc33c4cacee8
fe7b21b",
    "timestamp": "2023-03-12T21:29:16.508Z",
    "isDelete": false
  },
  {
    "record": {
      "Action": "CREATE",
      "User": "Doc1",
      "department": "Org1MSP",
      "MedicalRecord": {
        "TaxNumber": "TEST",
        "Smoking": "Yes",
        "AlcoholDrinking": "Yes",
        "DiffWalking": "Yes",
        "PhysicalActivity": "Yes",
        "Asthma": "Yes"
      }
    },
    "txId":
"f14aa49ec5f7a85123357238c7db40010c6227b3368b796d7b741eaf3
e4090a9",
```

Fig. 7. Result shown when an access control policy succeeds.

database, but this alternative is out of the scope of this work. The definition of the access control policies is made inside the functions defined in the smart contract. An example of access control policy refers to the reading and the updating of a medical record, as shown in the listing of code 9.

```
MSPID, _ := ctx.GetClientIdentity().GetMSPID()
   if asset.Organization != MSPID {
   return true, &AccessDenied{}
   }
```

Listing 9: Definition of the read and update access control policy

As another example, the listing of code 10 shows the access control policy definition and Fig. 7 the result obtained into the web app, in case a user with "Administrator" role requests to read the history of changes in a medical record.

```
role, _, _ := ctx.GetClientIdentity().
                  GetAttributeValue("Role")
   if role != "Administrator" {
     return nil, &AccessDenied{}
   }
```

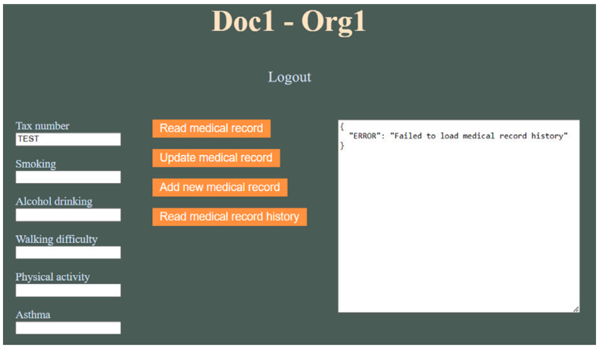
Listing 10: Definition of the read medical record history policy

On the other hand, if this check fails, then a message saying that the action failed appears, as shown in Fig. 8.

4.7. Discussion about the proposed approach

This Section focused on the definition of the blockchain elements and security aspects, as well as on the definition of smart contracts and access control policies. Note that data violation is prevented by coupling blockchain technologies with a clever management of access control over the resources' disclosure. However, the advantages of the presented approach also lie into the outcomes which could come from the analysis of the information retrieved by the blockchain.

In fact, the envisioned system allows to conduct an analysis of current available products and information flow for recall, expiration, shortages and counterfeits management processes in the healthcare supply chain [23]. Analyzing existing practice is useful to tracing process details about the tasks, and to identify gaps and the order in which information is processed and accessed by various stakeholders. Note that mixed lots delivery could have broader implications in healthcare supply chain leading to pricing discrepancies, poor inventory practices, and data errors in medical records. For such a reason, realizing a solution able to integrate both medical records and drugs information could support a more efficient management of drugs supply, avoiding waste of resources and facing more efficiently the needs of end-users.



```
{
  "ERROR": "Failed to load medical record history"
}
```

Fig. 8. Message shown when an access control policy fails.

Recall management in healthcare requires timely and accurate information communication across all stakeholders in a healthcare supply chain, since the most common issue encountered by care providers and distributors is the absence of a searchable product purchase history. This has significant consequences on both the safety of patients and treatment outcomes. Another challenge for care providers includes receiving recall communication at the wrong clinical or business unit and, in many cases, not receiving any recall information at all. Moreover, no standard practice currently exists across the healthcare supply chain to handle products' expiration. Effectively managing outdated products and expiration is critical to eliminate supply chain process redundancy and reduce waste. Unfortunately, today, a common strategy to deal with products' expiration is to increase the number of purchase orders and quantity to ensure product availability, and, although manufacturers are required to maintain a log of distributed products, tracking for expiration beyond a manufacturer's dispatch is not recorded nor shared among the stakeholders participating in the supply chain. On the other hand, product shortages in healthcare adversely affect treatment outcomes and patient safety. In such a situation, stakeholders should have the responsibility to find substitute products quickly. Therefore, a system that enables proactive information exchange and inventory visibility among stakeholders is fundamental. To such scopes, the network proposed in this paper, should help in keeping trace of the transactions related to products, as well as on the market demand.

Finally, the adoption of blockchain technology should mitigate the social issues related to the acceptance by users and companies of such tracking and traceability systems. A further improvement would consist in the integration with more specific provenance search and tracking mechanisms, as the one proposed in [30], in order to efficiently managing traceability and auditability and also investigating possible attacks. In fact, the security functionalities offered by the blockchain help in managing and detecting counterfeits, but further mechanisms must be put in act in order to detect and face occurring attacks. Blockchain also presents some limitations, mainly due to high energy consumption, processing capacity, and storage problems. Hence, in the definition of our solution, we considered the fact that blockchain will run in a distributed environment and, as assessed in Section 5, in IoT powerful smart devices; a potentially single-point-of-failure as a centralized cloud is not considered.

5. Performance evaluation

In order to assess the feasibility and infer about the scalability of the envisioned system, we first propose the setup of a simple yet real test-bed, able to execute the blockchain-based framework in a distributed manner, and to simulate the interactions with IoT devices. As regards interoperability and adaptability, we have that: (i) inside the blockchain framework, well-defined APIs have been specified in Section 4. Such APIs allow the information exchanges among the peers, the ordering services and the web app running on the clients; (ii) the client also provides APIs as communication interfaces with users and stakeholders, in order to acquire their requests in a standard format. Hence, in order to communicate with existing healthcare IT systems, we will need to adapt such APIs and the messages' format in order to gather the information of interest for the involved parties; (iii) end-IoT devices, which communicate directly with users or stakeholders, in a real scenario will use the protocols related to the particular technology adopted (e.g., RFID, wireless sensor, wearable device, etc.). Note that the tests conducted in this paper include the part of the network which is related to the blockchain, while the interaction with real end-IoT devices and parties is left as a future work, possibly in cooperation with interested healthcare structures, in order to start collaborating and setting up pilot projects.

Performance has been evaluated focusing on the resources' consumption and the time required to execute the various operations. Simulations have been firstly conducted on a machine with the following features:

- **CPU**: Intel(R) Core(TM) i7-9750H CPU @ 2.60 GHz 2.59 GHz
- **RAM**: 16 GB
- **Operating system**: Windows 10, 64 bit

Then, interactions with requests/transactions from users and peers have been moved to IoT powerful smart devices, namely Raspberry Pi, in order to produce a small test-bed for comparison with a more decentralized scenario, as would happen in a real distributed IoT environment, which leverages the decentralized potentialities of blockchain technology. In this sense, the IoT drives and integrates the blockchain features. A centralized solution does not apply in real scenarios, where different stakeholders are normally involved. For such a reason, in this paper, the assessment of performance will help in understanding the features (mainly with respect to hardware than software) of the technologies which could be adopted for achieving a totally distributed healthcare supply chain system. Raspberry Pi 3, model B, have the following features:

- **CPU**: Quad Core 1.2 GHz Broadcom BCM2837 64 bit CPU
- **RAM**: 1 GB
- **Operating system**: Raspberry Pi OS (previously called Raspbian)

Table 1 summarizes the configurations used for conducting simulations. Note that different parameters can be varied, starting from the number of records managed by the blockchain, to the requests coming from users in a certain time interval (i.e., one second). Also, the number of organizations, the channels per organizations and the peers belonging to an organization can be changed to assess the behavior of the network. Hence, three scenarios, detailed in Table 1, have been individuated to conduct the analysis. Such entities, along with the defined configurations, have been implemented to run either on a single personal computer or on the test-bed. In this second case, organizations, channels, and peers are equally split among three different Raspberry Pi, as shown in Fig. 9.

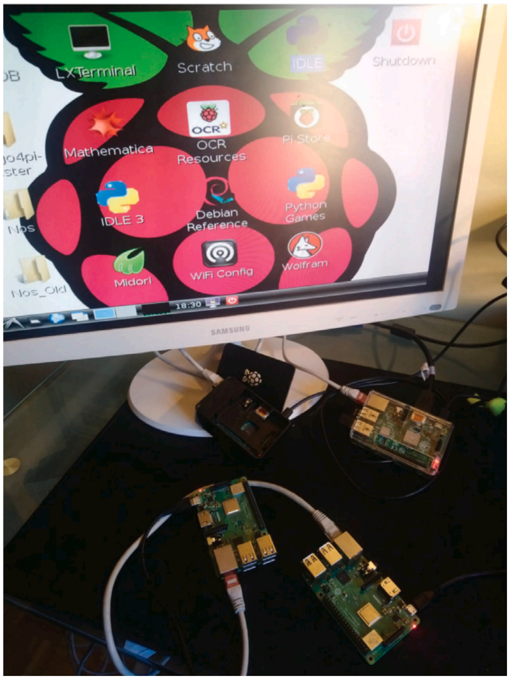


Fig. 9. Test-Bed representation.

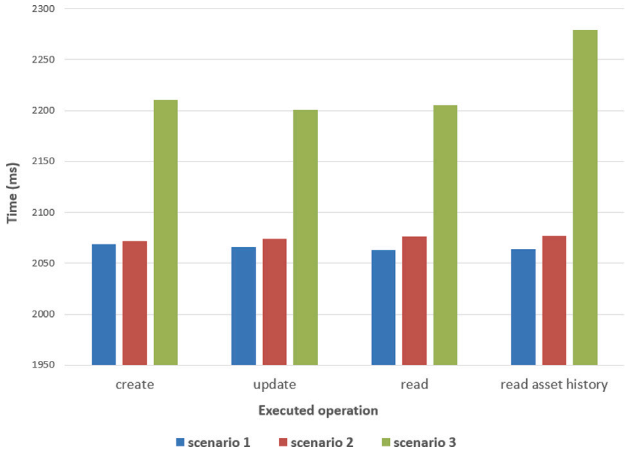


Fig. 10. Time required to execute the operations on the ledger — personal computer.

Table 1
Simulation parameters and configurations.

Parameters	Scenario 1	Scenario 2	Scenario 3
Num. of records/assets	100	1,000	10,000
Requests/sec	100	100	100
Number of organizations	3	5	5
Number of channels per organization	1	3	3
Number of peers per organization	10	100	100

Fig. 10 shows the average time required to create, read, update, and read the history of an asset, in case the whole system runs on the personal computer described above. The time has been evaluated with different numbers of assets stored in the blockchain, that are 100, 1,000, and 10,000. In all cases, the time required is always around 2 s. By increasing the number of assets in the blockchain, the average value is always around 2 s, with a slight increase of around 200 ms in the worst case tested (i.e., 10,000 assets). Therefore, by increasing the number of assets, the performances worsen, but of a negligible value.

Instead, Fig. 11 shows the same results when the transactions are executed through the mentioned test-bed. Performances are similar for the first and the second scenario, while they get worse in the third one. This is due to the difference in the CPU frequency, which is lower for the Raspberry Pi, as well as the quantity of RAM. Hence,

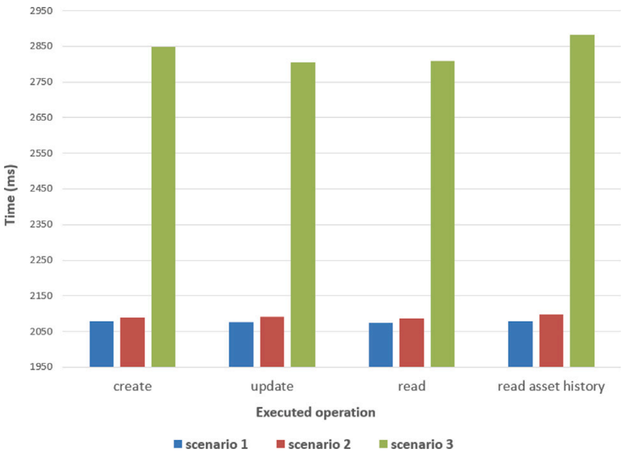


Fig. 11. Time required to execute the operations on the ledger — test-bed.

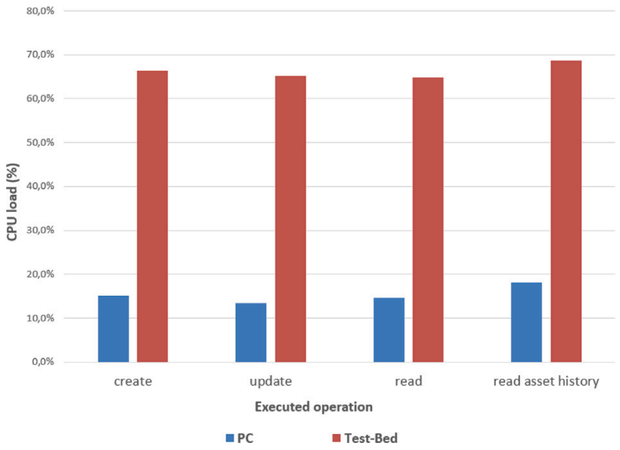


Fig. 12. Average CPU load — scenario 3.

to efficiently distributing blockchain services, creating a sort of fog computing layer [31], IoT devices must be equipped with reasonable hardware resources, depending on the workload to be managed. To justify such a conclusion, Figs. 12 and 13 show the average percentage of CPU load and RAM used by the simulated system in the third scenario, in case of execution either on the personal computer or on the dedicated test-bed (in this last case, the average is calculated on the three Raspberry Pi, where each Raspberry owns a maximum of 1 GB of RAM).

Finally, network throughput (i.e., the number of successful transactions per second) has been analyzed, as shown in Figs. 14 and 15. Obtained results are compliant with the outcomes related to the execution time, since the same issues arise in the comparison with the simulation on the personal computer and on the test-bed.

6. Conclusions

In this paper, an IoT-driven blockchain-based architecture to manage the healthcare supply chain and protect medical records from tampering and access violation, has been proposed. The permissioned blockchain Hyperledger Fabric has been adopted, due to the sensitive and private nature of the collected data. Proper access control policies have been defined to regulate the access to resources. In particular, both organization's membership and role inside the healthcare environment have been taken into account in the smart contract's definition. Performance assessment revealed that the proposed solution could be suitable in a large-scale environment, while more experiments must be done in the next future. As a future research direction, the integration of IoT-driven blockchain and artificial intelligence (AI) can revolutionize the healthcare industry [32]. Some available solutions already make use of neural networks to conduct anomaly detection activities inside a smart healthcare scenario [33], or employ contextual information/features along with machine learning techniques to formulate decisions on the basis of the current state of the system [34] or of people

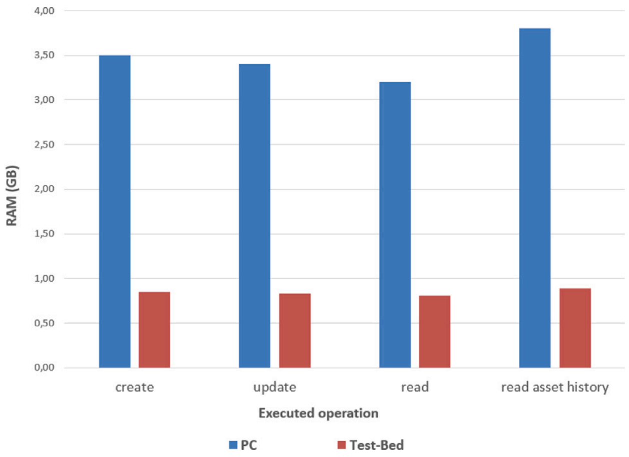


Fig. 13. Average RAM occupancy — scenario 3.

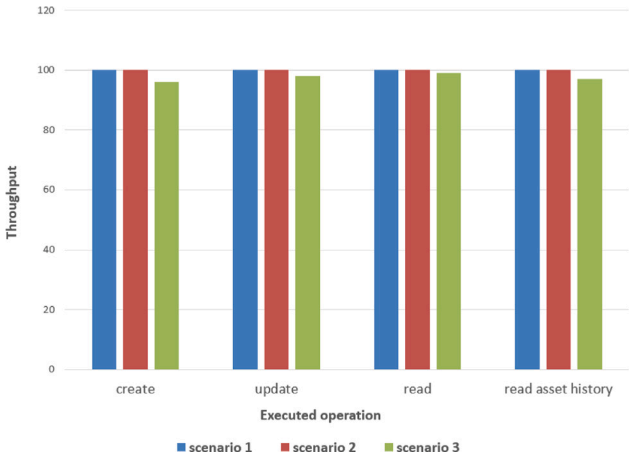


Fig. 14. Network throughput — personal computer.

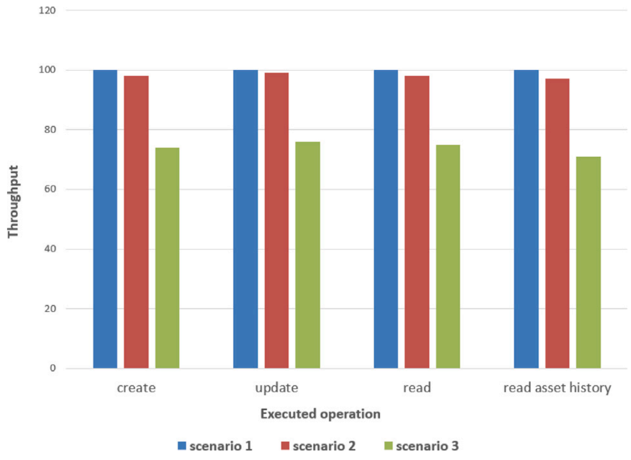


Fig. 15. Network throughput — test-bed.

(e.g., by analyzing diagnostic images) [35–38]. The identification of malicious devices along with a continuous trust assessment monitoring would improve the reliability of the whole system. Also, blockchain may facilitate the resolution of consistency and accuracy problems associated with healthcare data by further enhancing them using AI techniques.

CRediT authorship contribution statement

Alessandra Rizzardi: Writing – review & editing, Software, Methodology, Funding acquisition, Conceptualization. **Sabrina Sicari:** Writing – review & editing, Visualization, Supervision. **Jesus F. Cevallos M.:** Writing – original draft, Software. **Alberto Coen-Porisini:** Writing – review & editing, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work was supported in part by the SERENA-IIoT project, which has been funded by MUR (*Ministero dell'Universitá e della Ricerca*) under the PRIN 2022 program (project code 2022CN4EBH), and in part by project SERICS (project code PE00000014), under the NRRP MUR program funded by the EU - NGEU.

References

[1] Sreelakshmi Krishnamoorthy, Amit Dua, Shashank Gupta, Role of emerging technologies in future IoT-driven healthcare 4.0 technologies: A survey, current challenges and future directions, J. Ambient Intell. Humaniz. Comput. 14 (1) (2023) 361–407.

[2] Mehrdokht Pournader, Yangyan Shi, Stefan Seuring, S.C. Lenny Koh, Blockchain applications in supply chains, transport and logistics: A systematic review of the literature, Int. J. Prod. Res. 58 (7) (2020) 2063–2081.

[3] Ritesh Patel, Milena Migliavacca, Marco E. Oriani, Blockchain in banking and finance: A bibliometric review, Res. Int. Bus. Finance 62 (2022) 101718.

[4] Konstantinos Demestichas, Nikolaos Peppes, Theodoros Alexakis, Evgenia Adamopoulou, Blockchain in agriculture traceability systems: A review, Appl. Sci. 10 (12) (2020) 4113.

[5] Anton Hasselgren, Katina Kralevska, Danilo Gligoroski, Sindre A. Pedersen, Arild Faxvaag, Blockchain in healthcare and health sciences - a scoping review, Int. J. Med. Inf. 134 (2020) 104040.

[6] Anushree Tandon, Puneet Kaur, Matti Mäntymäki, Amandeep Dhir, Blockchain applications in management: A bibliometric analysis and literature review, Technol. Forecast. Soc. Change 166 (2021) 120649.

[7] P. Chinnasamy, P. Deepalakshmi, HCAC-EHR: Hybrid cryptographic access control for secure EHR retrieval in healthcare cloud, J. Ambient Intell. Humaniz. Comput. 13 (2) (2022) 1001–1019.

[8] Oladayo Olufemi Olakanmi, Kehinde Oluwasesan Odeyemi, Expressible access control scheme for data sharing and collaboration in cloud-centric internet of medical things system, J. Ambient Intell. Humaniz. Comput. 14 (6) (2023) 7189–7205.

[9] Moulouki Reda, Dominique.Bernard Kanga, Taif Fatima, Mohamed Azouazi, Blockchain in health supply chain management: State of art challenges and opportunities, Procedia Comput. Sci. 175 (2020) 706–709.

[10] Aashima Sharma, Sanmeet Kaur, Maninder Singh, A comprehensive review on blockchain and internet of things in healthcare, Trans. Emerg. Telecommun. Technol. 32 (10) (2021) e4333.

[11] Faisal Jamil, Shabir Ahmad, Naeem Iqbal, Do-Hyeun Kim, Towards a remote monitoring of patient vital signs based on iot-based blockchain integrity management platforms in smart hospitals, Sensors 20 (8) (2020) 2195.

[12] Vahiny Sharma, Ankur Gupta, Najam Ul Hasan, Mohammad Shabaz, Isaac Ofori, Blockchain in secure healthcare systems: State of the art, limitations, and future directions, Secur. Commun. Netw. (2022) 2022.

[13] Rui Zhang, Rui Xue, Ling Liu, Security and privacy for healthcare blockchains, IEEE Trans. Serv. Comput. 15 (6) (2021) 3668–3686.

[14] Abdurrashid Ibrahim Sanka, Ray C.C. Cheung, A systematic review of blockchain scalability: Issues, solutions, analysis and future researchs, IEEE J. Netw. Comput. Appl. 195 (2021) 103232.

[15] Jin Ho Park, Jong Hyuk Park, Blockchain security in cloud computing: Use cases, challenges, and solutions, Symmetry 9 (8) (2017) 164.

[16] Qi Xia, Emmanuel.Boateng Sifah, Abla Smahi, Sandro Amofa, Xiaosong Zhang, Bbds: Blockchain-based data sharing for electronic medical records in cloud environments, Information 8 (2) (2017) 44.

[17] Xiao Yue, Huiju Wang, Dawei Jin, Mingqiang Li, Wei Jiang, Healthcare data gateways: Found healthcare intelligence on blockchain with novel privacy risk control, J. Med. Syst. 40 (2016) 1–8.

[18] Safa Shubbar, Ultrasound Medical Imaging Systems using Telemedicine and Blockchain for Remote Monitoring of Responses to Neoadjuvant Chemotherapy in Women's Breast Cancer: Concept and Implementation (Ph.D. thesis), Kent State University, 2017.

[19] Cleverence Kombe, Mussa Dida, Anael Sam, A Review on Healthcare Information Systems and Consensus Protocols in Blockchain Technology, 2018.

[20] Ahmad Musamih, Khaled Salah, Raja Jayaraman, Junaid Arshad, Mazin Debe, Yousof Al-Hammadi, Samer Ellahham, A blockchain-based approach for drug traceability in healthcare supply chain, IEEE Access 9 (2021) 9728–9743.

[21] Ilhaam A. Omar, Raja Jayaraman, Mazin S. Debe, Khaled Salah, Ibrar Yaqoob, Mohammed Omar, Automating procurement contracts in the healthcare supply chain using blockchain smart contracts, IEEE Access 9 (2021) 37397–37409.

[22] Samin Sadri, Aamir Shahzad, Kaiwen Zhang, Blockchain traceability in healthcare: Blood donation supply chain, in: 2021 23rd International Conference on Advanced Communication Technology, ICACT, IEEE, 2021, pp. 119–126.

[23] Raja Jayaraman, Khaled Salah, Nelson King, Improving opportunities in healthcare supply chain processes via the internet of things and blockchain technology, in: Research Anthology on Blockchain Technology in Business, Healthcare, Education, and Government, IGI Global, 2021, pp. 1635–1654.

[24] Mueen Uddin, M.S. Memon, Irfana Memon, Imtiaz Ali, Jamshed Memon, Maha Abdelhaq, Raed Alsaqour, Hyperledger fabric blockchain: Secure and efficient solution for electronic health records, Comput. Mater. Contin. 68 (2) (2021) 2377–2397.

[25] Mueen Uddin, Blockchain medledger: Hyperledger fabric enabled drug traceability system for counterfeit drugs in pharmaceutical industry, Int. J. Pharm. 597 (2021) 120235.

[26] Victoria Ahmadi, Sophia Benjelloun, Michel El Kik, Tanvi Sharma, Huihui Chi, Wei Zhou, Drug governance: Iot-based blockchain implementation in the pharmaceutical supply chain, in: 2020 Sixth International Conference on Mobile and Secure Services, MobiSecServ, IEEE, 2020, pp. 1–8.

[27] Saroj Kumar Nanda, Sandeep Kumar Panda, Madhabananda Dash, Medical supply chain integrated with blockchain and iot to track the logistics of medical products, Multimedia Tools Appl. (2023) 1–23.

[28] Siamak Solat, Philippe Calvez, Farid Naït-Abdesselam, Permissioned vs. permissionless blockchain: How and why there is only one right choice, J. Softw. 16 (3) (2021) 95–106.

[29] Diego Ongaro, John Ousterhout, In Search of an UnderstandAble Consensus Algorithm. USENIX ATC'14, USENIX Association, USA, 2014, pp. 305–320.

[30] Xinyu Yang, Haoyuan Liu, Ziyu Wang, Peng Gao, Zebra: Deeply integrating system-level provenance search and tracking for efficient attack investigation, 2022, arXiv preprint, arXiv:2211.05403.

[31] Sabrina Sicari, Alessandra Rizzardi, Alberto Coen-Porisini, Insights into security and privacy towards fog computing evolution, Comput. Secur. (2022) 102822.

[32] Priti Tagde, Sandeep Tagde, Tanima Bhattacharya, Pooja Tagde, Hitesh Chopra, Rokeya Akter, Deepak Kaushik, Md Habibur Rahman, Blockchain and artificial intelligence technology in e-health, Environ. Sci. Pollut. Res. 28 (2021) 52810–52831.

[33] Ziyu Wang, Nanqing Luo, Pan Zhou, GuardHealth: Blockchain empowered secure data management and graph convolutional network enabled anomaly detection in smart healthcare, J. Parallel Distrib. Comput. 142 (2020) 1–12.

[34] Hamidreza Alikhani, Ziyu Wang, Anil Kanduri, Pasi Lilieberg, Amir M. Rahmani, Nikil Dutt, SEAL: Sensing efficient active learning on wearables through context-awareness, in: IEEE Design, Automation & Test in Europe Conference & Exhibition, DATE, 2024, pp. 1–2.

[35] Yunlong Zhang, Chenxin Li, Xin Lin, Liyan Sun, Yihong Zhuang, Yue Huang, Xinghao Ding, Xiaoqing Liu, Yizhou Yu, Generator versus segmentor: Pseudo-healthy synthesis, in: 24th International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI), vol. 142, 2021, pp. 150–160.

[36] Liyan Sun, Chenxin Li, Xinghao Ding, Yue Huang, Zhong Chen, Guisheng Wang, Yizhou Yu, John Paisley, Few-shot medical image segmentation using a global correlation network with discriminative embedding, Comput. Biol. Med. 140 (2022) 105067.

[37] Chenxin Li, Xin Lin, Yijin Mao, Wei Lin, Qi Qi, Xinghao Ding, Yue Huang, Dong Liang, Yizhou Yu, Domain generalization on medical imaging classification using episodic training with task augmentation, Comput. Biol. Med. 141 (2022) 105144.

[38] Ziyu Wang, Zhongqi Yang, Iman Azimi, Amir M. Rahmani, Differential private federated transfer learning for mental health monitoring in everyday settings: A case study on stress detection, 2024, arXiv preprint, arXiv:2402.10862.

Alessandra Rizzardi is Assistant Professor at University of Insubria (Varese), where she received BS/MS degree in Computer Science 110/110 cum laude in 2011 and 2013, respectively. In 2016 she got Ph.D. in Computer Science and Computational Mathematics at the same university. She is the Principal Investigator for the funded national project PRIN 2022, and Editorial Board member for the journals Transactions on Emerging Telecommunications Technologies (Wiley), Internet Technology Letters (Wiley), and Sensors (MDPI). Her research activity is on WSN and IoT security issues.

Sabrina Sicari is Full Professor at University of Insubria (Varese). She received degree in Electronical Engineering, 110/110 cum laude, from University of Catania, in 2002, where in 2006 she got Ph.D. in Computer and Telecommunications Engineering, followed by Prof. Aurelio La Corte. She is member of COMNET, IEEE IoT, ETT, ITL editorial board. Her research activity security, privacy and trust in WSN, WMSN, IoT, and distributed systems. She is IEEE senior member.

Jesús F. Cevallos M. received a Ph.D. in Computer Science Engineering from Sapienza University (Rome) in 2022. He now covers a post-doc researcher position at the University of Insubria (Varese). His main research interests are industrial/ethical applications of Deep Learning with a special focus on Natural Language Processing and Neural Algorithmic Reasoning.

Alberto Coen Porisini received Dr. Eng. degree and Ph.D. in Computer Engineering from Politecnico di Milano in 1987 and 1992. He is Full Professor of Software Engineering at Universitá degli Studi dell’Insubria since 2001, Dean of the School of Science from 2006 and Dean from 2012 to 2018. His research regards specification/design of real-time systems, privacy models and WSN.