IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 7 November 2022, accepted 29 January 2023, date of publication 3 February 2023, date of current version 9 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3242605

RESEARCH ARTICLE

Blockchain-Based Service-Oriented Architecture for Consent Management, Access Control, and Auditing

ISABEL ROMÁN-MARTÍNEZ[1,2], **JORGE CALVILLO-ARBIZU**[1,3], **VICENTE J. MAYOR-GALLEGO**[1,2], **GERMÁN MADINABEITIA-LUQUE**[1,2], **ANTONIO J. ESTEPA-ALONSO**[1,2], **AND RAFAEL M. ESTEPA-ALONSO**[1,2]

[1]Departamento de Ingeniería Telemática, Escuela Técnica Superior de Ingeniería, 41092 Seville, Spain
[2]Grupo de Ingeniería Telemática, Universidad de Sevilla, 41092 Seville, Spain
[3]Grupo de Ingeniería Biomédica, Universidad de Sevilla, 41092 Seville, Spain

Corresponding author: Jorge Calvillo-Arbizu (jcalvillo@us.es)

ABSTRACT Continuity of care requires the exchange of health information among organizations and care teams. The EU General Data Protection Regulation (GDPR) establishes that subject of care should give explicit consent to the treatment of her personal data, and organizations must obey the individual's will. Nevertheless, few solutions focus on guaranteeing the proper execution of consents. We propose a service-oriented architecture, backed by blockchain technology, that enables: (1) tamper-proof and immutable storage of subject of care consents; (2) a fine-grained access control for protecting health data according to consents; and (3) auditing tasks for supervisory authorities (or subjects of care themselves) to assess that healthcare organizations comply with GDPR and granted consents. Standards for health information exchange and access control are adopted to guarantee interoperability. Access control events and the subject of care consents are maintained on a blockchain, providing a trusted collaboration between organizations, supervisory authorities, and individuals. A prototype of the architecture has been implemented as a proof of concept to evaluate the performance of critical components. The application of subject of care consent to control the treatment of personal health data in federated and distributed environments is a pressing concern. The experimental results show that blockchain can effectively support sharing consent and audit events among healthcare organizations, supervisory authorities, and individuals.

INDEX TERMS Blockchain, consent management, fast healthcare information resources (FHIR), general data protection regulation (GDPR), service-oriented architecture (SOA), business process management (BPM).

I. INTRODUCTION

The EU General Data Protection Regulation (GDPR) establishes a new framework for personal data treatment while enhancing citizens' rights [1]. Personal health data are classified as a special category, the processing of which is strictly prohibited unless the data subject (identifiable Subject of Care, SoC) gives explicit consent, usually before collecting data.

Consents relate health data resources with purposes of use, stakeholders, and conditions for data treatment. The impact of SoC consents on healthcare organizations is twofold: they must observe consent before allowing the use or disclosure of personal health data, and consents can be used in auditing tasks by supervisory authorities to assess if data treatment performed by organizations is compliant with SoC's will. Although healthcare organizations are expected to comply with GDPR, supervisory authorities appointed by EU State Members could benefit from a transparent and accessible record of access control logs and consents to detect violations of SoC consent.

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

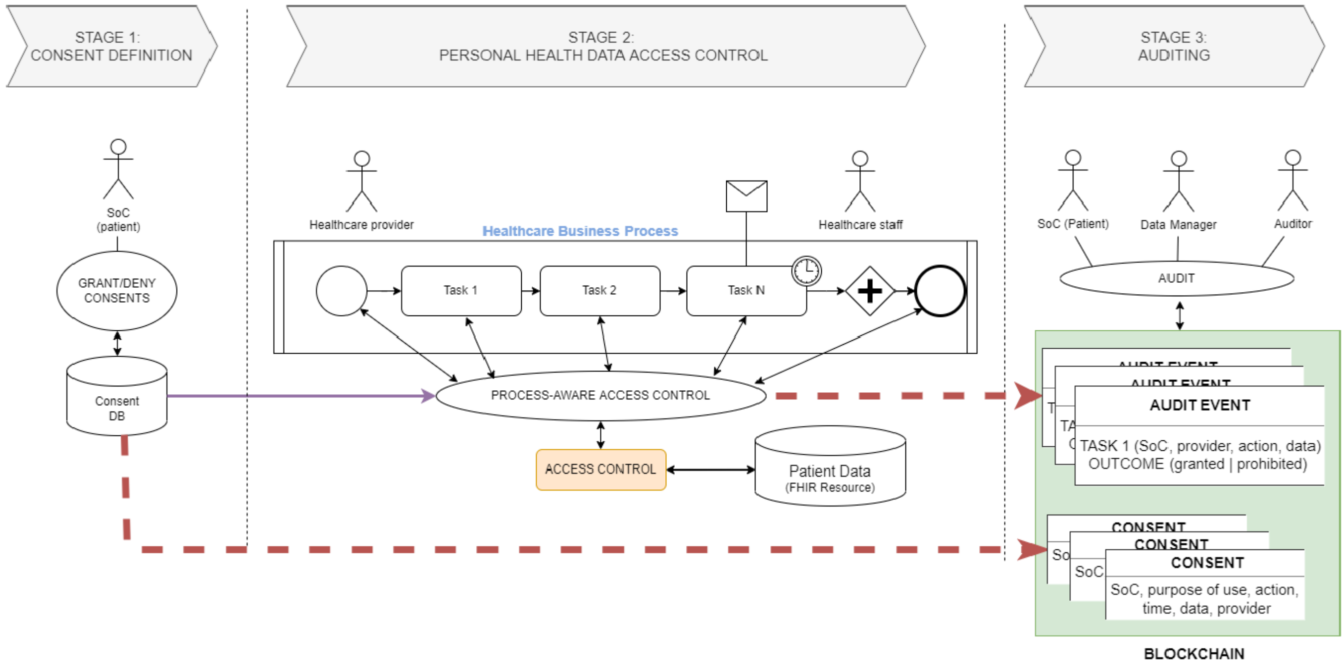

STAGE 1:
CONSENT DEFINITION
STAGE 2:
PERSONAL HEALTH DATA ACCESS CONTROL
STAGE 3:
AUDITING
SoC (patient)
GRANT/DENY CONSENTS
Consent DB
Healthcare provider
Healthcare Business Process
Healthcare staff
Task 1
Task 2
Task N
PROCESS-AWARE ACCESS CONTROL
ACCESS CONTROL
Patient Data (FHIR Resource)
SoC (Patient)
Data Manager
Auditor
AUDIT
AUDIT EVENT
TASK 1 (SoC, provider, action, data)
OUTCOME (granted | prohibited)
CONSENT
SoC, purpose of use, action, time, data, provider
BLOCKCHAIN

FIGURE 1. Scope of the proposed scheme.

Since a GDPR-based auditing mechanism requires combining exhaustive information about access to health data resources and SoC consents, the integrity of both data elements is paramount.

This work presents a service-oriented architecture (SOA), combined with blockchain technology, supporting: (1) authorship and immutability of SoC's consents and audit logs; (2) privacy of personal health data through a fine-grained access control service where access policies align with consents; and (3) auditing capabilities to assess compliance of healthcare organizations with GDPR and SoC's desires. The software architecture proposed in this work is based on standards and widely used paradigms such as SOA and Business Process Management (BPM), which enhances the interoperability of the approach and its deployment in distributed or federated environments where different actors/organizations participate.

Fig. 1 illustrates the scope of this work as a high-level process. Initially, the SoC defines consents that are stored in a database and persisted in the blockchain (Stage 1). Consents are related to personal health data that may be requested during the execution of healthcare business processes (e.g., in a physical encounter between a SoC and a clinician) (Stage 2). A business process-aware access control service will be responsible for granting (or not) access to data according to consents. The access decision on each data request will be registered as an audit event that will also be persisted in the blockchain. In due course (Stage 3), supervisory authorities, SoCs, or healthcare organizations, as part of their business activity monitoring, may access the blockchain to recover audit information and consents for assessing the access control service decisions.

After GDPR and HIPAA directives, dynamic consent has become a hot topic in the literature [2]. In the healthcare domain, several approaches solve consent management through blockchain, a technology trend in this field [3]. For instance, a general-purpose blockchain platform that adopts an ontology-based consent model has been developed for managing data subject consents [4]. In [5], Rantos et al. present a blockchain platform that facilitates data controller interaction with the data subject under GDPR. Data controllers use the platform to inform the user transparently about any personal data processing, including the purpose of use, the period, and the legal basis of processing. Tith et al. also suggest using blockchain to store patient data access consent [6]. Consents are based on the purpose of use, which limits the granularity of the access control mechanism. Table 1 shows some relevant schemes available in the literature. They are compared with our work. The main drawbacks of reviewed approaches are the lack of a standardized consent model for enhancing interoperability, limited access control mechanisms such as control listing or role-based models, disregarding auditing tasks, or storage of health data on the blockchain.

Considering the related literature, our approach presents several innovative contributions:

1. Blockchain-based support for auditing services that could be used by supervisory authorities or SoCs for assessing GDPR compliance. Using blockchain brings an immutable and tamper-proof ledger for consent and access control decisions for auditing.
2. Fine-grained backward access control model and service to protect personal health data in healthcare SOA.

TABLE 1. Comparison of the approached system with related works.

Ref.	GDPR	Dynamic Consent	Consent Format	Access Control/Model	Audit	Field	Blockchain platform	Type of network	Blockchain content	Implementation	Performance evaluation
[4]	Yes	Yes	Ontology	Yes/ XACML	No	General	Quorum	Permissioned	Dataset profiles	Yes	Yes
[5]	Yes	Yes	Ontology	Yes/ XACML	No	IoT in health	Ethereum	Public	Consents	Yes	No
[6]	No	Yes	Proprietary	Yes /Purpose-based	Yes	Health	H. Fabric	Permissioned	Consents	Yes	No
[7]	Yes	Yes	FHIR-based	Yes/ ABAC	No	Health	-	-	-	No	No
[8]	Yes	Yes	Ontology	Yes/ Purpose-based	Yes	Health	Ethereum	Public	Health data Consents	Yes	Yes
[9]	Yes	No	-	Yes/-	No	Health	H. Fabric	Permissioned	Health data storage	Yes	No
[10]	Yes	Yes	-	Yes /RBAC	Yes	Clinical trials	H. Fabric	Permissioned	Consents	Yes	No
[11]	Yes	Yes	-	Yes/-	Yes	Medical research	H. Fabric	Permissioned	Consents and Access logs	Yes	Yes
[12]	No	Yes	Proprietary	No	No	Health	H. Fabric	Permissioned	Hash data	Yes	No
Our approach	Yes	Yes	FHIR	Yes/ XACML	Yes	Health	H. Fabric	Permissioned	Consents and AuditEvents	Yes	Yes

3. Standard-based approach (FHIR, XACML/SAML…) that guarantees interoperability and openness in distributed environments.
4. Experimental analysis to identify bottlenecks or delays in the application of blockchain for auditing.

The novelty of this work lies in the combination of technologies, paradigms and especially, standards, to address the inherent complexity of the GDPR within the healthcare domain. As has been described, there are approaches in this field, but none has proposed a complete resolution to the consent management business process. Thus, the innovative contribution of our work is to deal with the entire chain of the consent management process (beyond collection and storage).

Due to the complexity of the proposed solution (which operates both at the process and data levels and combines different technologies), this work aims to provide a high-level description rather than explaining implementation details. Nonetheless, the critical points of the solution (e.g., blockchain) are thoroughly explained, analyzed, and validated.

The structure of this paper is organized as follows. Section II outlines basic ideas on GDPR, FHIR, blockchain, SOA, and BPM. A detailed description of our approach is shown in Section III, and Section IV presents a proof of concept and experimental results. Section V discusses results and, finally, Section VI covers the conclusions and limitations of the work.

II. MATERIALS AND METHODS

A. EU GENERAL DATA PROTECTION REGULATION (GDPR)

GDPR was designed to protect the privacy of EU citizens' data and to reshape the way organizations manage personal data (i.e., any information relating to an identifiable natural person) [1]. GDPR restricts the treatment of personal data (including collection, recording, organization, structuring, storage, adaptation or alteration, retrieval, consultation, use, disclosure by transmission, dissemination or otherwise making available, alignment or combination, restriction, erasure, or destruction of data) unless data subject gives explicit consent.

Within this framework, two main entities are identified: the supervisory authority and the controller. The former is an independent public authority established by an EU State Member and responsible for monitoring the application of this regulation and, consequently, for protecting the fundamental rights and freedoms of natural persons. The latter is appointed in each organization as responsible for the treatment of personal data. The controller shall send the record of processing activities to the supervisory authority on request. In the case of personal data breach, GDPR follows a trust model where the controller is expected to notify the competent supervisory authority without undue delay.

Our approach aims to enhance the privacy guarantees of SoCs and reduce the odds of an organization covering data violations. To do this, SoC consents and data treatment events are shared via a blockchain ledger, tampered-proof and accessible by the supervisory authority for auditing. The integrity and trustworthiness of data stored in blockchain allow the supervisory authority to identify the misuse of personal health data proactively. Furthermore, the SoC could be aware of the treatment of her personal health data by checking data in the blockchain.

B. ACCESS CONTROL SERVICES IN HEALTHCARE SOA

The wide variety of user profiles, purposes of use, and information resources in the health domain create challenging requirements for access control. Additionally, the distributed nature of organizations, information, and accesses requires deploying distributed authorization models. The standard ITU-T X.812 [13] establishes an access control framework applicable to any domain and based on distributed components, each performing a particular task. Fig. 2 shows this framework using components naming from XACML [14]:

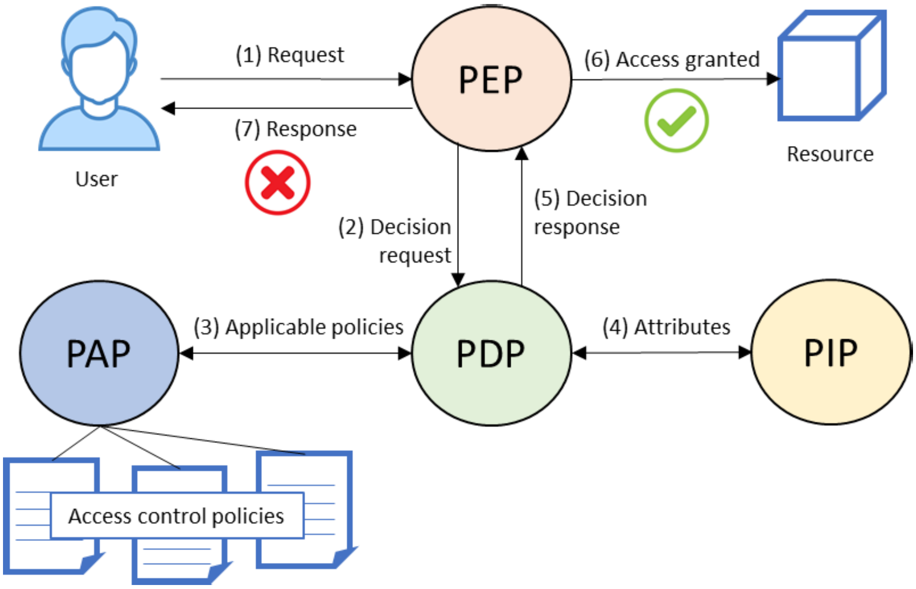


FIGURE 2. Access control model adapted from XACML [14].

- Policy Enforcement Point (PEP): acts as a request interceptor (1) and sends it to PDP (2).
- Policy Decision Point (PDP): receives a request from PEP (2), gathers policies and information applicable to the request from PAP (3) and PIP (4), respectively, evaluates policies, and makes an access decision (i.e., granted or prohibited) (5).
- Policy Administration Point (PAP): manages access control policies.
- Policy Information Point (PIP): stores information required for policy evaluation (e.g., data about users, resources, or context).

Two open standards ease the implementation of the ITU-T X.812 framework in SOA: the Security Assertion Markup Language (SAML) [15] and the eXtensible Access Control Markup Language (XACML) [14]. While SAML defines a protocol for exchanging authentication and authorization data between parties, XACML specifies a fine-grained, attributed-based access control (ABAC) policy language, and a processing model describing how to evaluate access requests according to the rules defined in policies.

We adopt the ITU-T X.812 framework and the ABAC access control model [16] due to its fine-grained access control (needed for protecting FHIR resources, see next Section) versus other models, such as RBAC or DAC [17]. SoC explicit consents are translated to access control policies in XACML format. These policies, combined with organizational ones, will support the authorization process performed by the PDP in response to any attempt to access healthcare data.

C. FAST HEALTHCARE INFORMATION RESOURCES (FHIR)

FHIR is a standard for health data exchange published by HL7 [18], and it is built upon the RESTful paradigm to guarantee interoperability across organizations and clinical teams. Healthcare information entities are managed as resources (e.g., Patient, Encounter, Observation…), the representation of which, at the instance level, is composed of mandatory and optional attributes. Additionally, a resource may refer others (e.g., an Observation resource includes links to related resources such as Patient and Encounter).

The FHIR API establishes a set of operations on resources known as interactions (get, update, delete…). A single interaction may involve multiple resources (e.g., all instances of type Observation related to a patient), resulting in a Bundle, a resource composed of several resources. Protecting access to resources requires a fine-grained access control solution to discriminate which resources in a Bundle may be sent to the requester and which may not.

Beyond health-related, other resources related to security or process management defined by the standard are also of interest to our approach. In particular, the *Consent* resource expresses agreements between a healthcare consumer (e.g., SoC) and an authorized entity (grantee) to permitted or prohibited actions with limitations on the purpose of use. The *AuditEvent* resource supports the registration of any kind of event as described by certain attributes (type, creation time, event time, authorship, history, status, event purpose, the action performed, event outcome…), the actors involved, and other resources used during the event [19]. Furthermore, the *Task* resource stores information about an instance of a process and includes references to all the resources generated during its execution.

D. BLOCKCHAIN

A blockchain is a shared, distributed ledger that records transactions and is maintained by multiple nodes in a network where nodes do not trust each other. Blockchain belongs to the so-called DLT (Distributed Ledger Technology) [20],

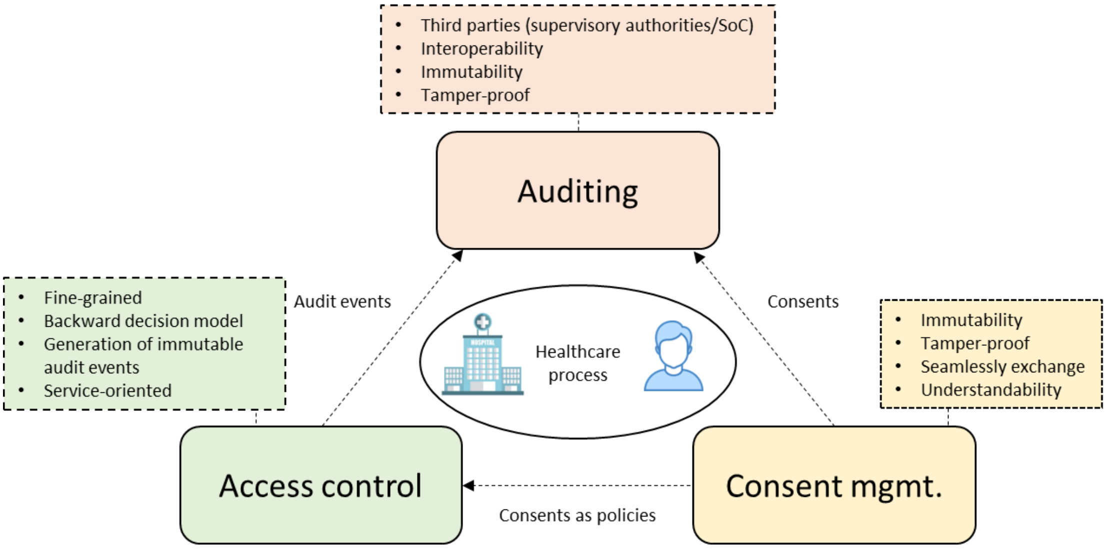


FIGURE 3. Requirements of security capabilities.

where each node has an identical copy of the ledger, represented as a chain of blocks, each block is a logical sequence of transactions. Each block encloses the hash of its immediate previous block, which guarantees the immutability of the ledger. Blockchain technologies provide serializability, immutability, and cryptographic verifiability without a single point of trust [21]. These properties have boosted blockchain adoption in various industries, including health.

We can distinguish two categories of blockchain: permissionless (such as Bitcoin [22] or Ethereum [23]), in which anyone can join the network to perform transactions, and permissioned network (e.g., Hyperledger Fabric [24]), suitable for enterprise applications that require authenticated participants. In a permissioned network, each node can be owned by a different organization.

In this work, a permissioned blockchain network is deployed to allow participating healthcare organizations and supervisory authorities to share:

1. Audit logs (i.e., attempts to access health resources by organizations).
2. Consents granted by SoC.

Supervisory authorities will be able to verify GDPR compliance and detect data breaches autonomously without the need to be notified by the organization's controller. Likewise, SoC will be able to analyze how her consents are being fulfilled and how her personal health data are being treated.

III. BLOCKCHAIN-BASED CONSENT-AWARE AUDITING SERVICE ARCHITECTURE

A. ARCHITECTURE REQUIREMENTS

An auditing service for assessing the protection of health resources custodied by organizations according to consents issued by data owners needs to rest upon two capabilities: consent management and access control. Fig. 3 shows the main design requirements for the three key elements of this work. These requirements shape the software architecture in the following aspects:

- Consent management: any healthcare provider accessing SoC's personal health data must be aware of related consents. Thus, consents should be seamlessly distributed, understandable by all parties, trustworthy, and immutable. Our proposal addresses interoperability issues by using the standard FHIR for modelling consents (*Consent* resource). At the same time, immutability and tamper-proof requirements are fulfilled through a dedicated communication channel in the blockchain where consents are stored in blocks shared by all parties involved.
- Access Control: an access control service must be in place during the operation of healthcare processes through the SOA environment. This service is responsible for granting (or not) access to data according to SoC's consents and generating auditable information. FHIR interactions may involve complex responses (e.g., a Bundle of resources) that advise the use of fine-grained policies and a backward decision model (i.e., actual data accessed in the SOA is only known in responses, not in requests). Therefore, access control needs to be service-oriented, fine-grained, backward, and capable of gathering all relevant information for later auditing.
- Auditing: a central piece of our proposal is a private, permissioned blockchain shared by healthcare organizations and supervisory authorities. Storing standardized audit events and consents in a shared ledger brings some advantages to the traditional centralized log. Firstly, it enables the traceability of consents and personal health data treatment by third parties. Secondly, the immutability of data blocks prevents organizations from modifying

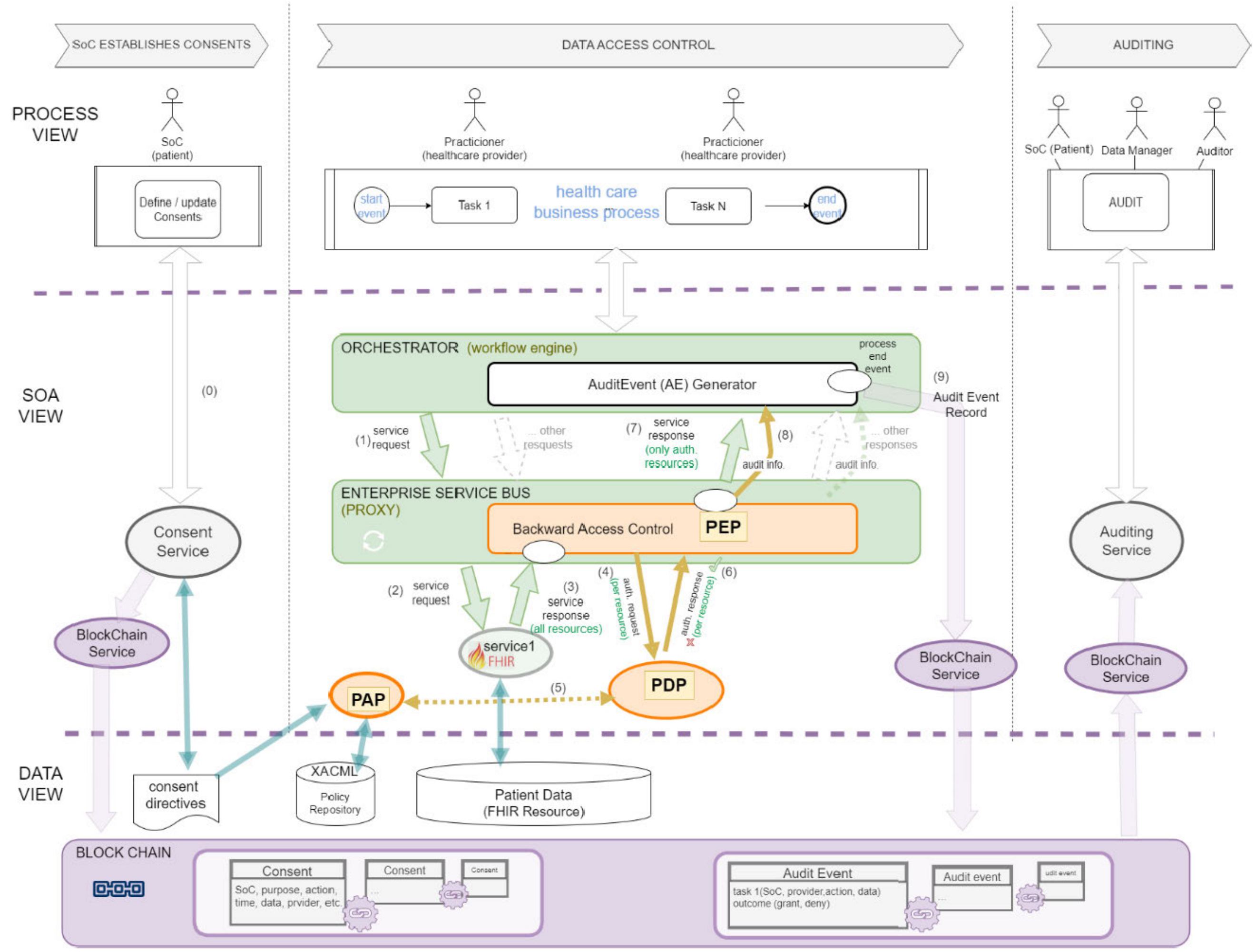


FIGURE 4. Architecture of the proposed audit service based on SOA and blockchain.

or deleting actions performed on their resources or tampering SoC consents, guaranteeing nonrepudiation in case of a data breach.

B. PROPOSED ARCHITECTURE

Our solution is rooted in two main entities: a BPM-driven SOA (deployed in each participating organization) that includes FHIR and access control services and a blockchain involving healthcare organizations, supervisory authorities, and potentially, SoCs. Fig. 4 illustrates the proposed architecture by following the storyline from Section I in three views: business processes, SOA, and data. The data flow of a request to a FHIR service is shown in the SOA view.

Our scheme is supported by the following inter-related elements:

1. Consent management service (leftmost part in Fig. 4).
2. BPM-driven SOA architecture for healthcare business process (green elements).
3. Backward access control service (orange color).
4. Generator of access logs as FHIR *AuditEvents* (grey elements in the Orchestrator).
5. A blockchain system for storing and auditing (purple elements).

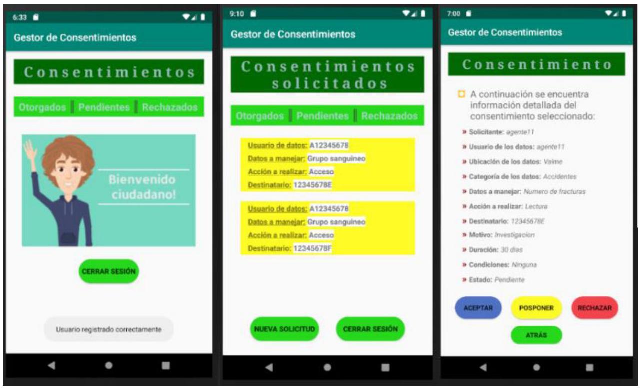
1) CONSENT MANAGEMENT SERVICE

Under GDPR, SoC should be asked for consent before personal health data treatment (Fig. 4, step 0). A consent management service has been deployed to collect consent from SoCs (Fig. 5). This service stores SoC's will as FHIR *Consent* resources in a central repository from where they can be accessed by the PAP component of the access control service.

Additionally, *Consents* are sent to the blockchain service to be available to other stakeholders. The implementation of the consent management service is out of the scope of this approach, and it is only referred here for clarity's sake of source of Consents.

2) BPM-DRIVEN SOA ARCHITECTURE FOR HEALTHCARE BUSINESS PROCESS

Healthcare organizations design and run business processes (i.e., patient discharge, referral orders, etc.) to fulfil their goals. Business processes are typically well-documented



Usuario de datos: A12345678
Datos a manejar: Grupo sanguineo
Acción a realizar: Acceso
Destinatario: 12345678E

Usuario de datos: A12345678
Datos a manejar: Grupo sanguineo
Acción a realizar: Acceso
Destinatario: 12345678F

A continuación se encuentra información detallada del consentimiento seleccionado:

» Solicitante: agente11
» Usuario de los datos: agente11
» Ubicación de los datos: Valme
» Categoría de los datos: Accidentes
» Datos a manejar: Numero de fracturas
» Acción a realizar: Lectura
» Destinatario: 12345678E
» Motivo: Investigacion
» Duración: 30 dias
» Condiciones: Ninguna
» Estado: Pendiente

FIGURE 5. Consent management service implemented as a mobile app.

thanks to notations such as the Business Process Model and Notation (BPMN) [25]. These models can be translated into execution processes [26], implemented over web service calls and handled by orchestrators (i.e., workflow engines in the SOA business process choreography layer [27]).

In our approach, the Orchestrator is aware of the tasks in the business model, the corresponding SOA service requests and responses, and the events of the execution process. We developed a component inside the Orchestrator called Audit Event (AE) Generator that collects information from the access control service regarding data access requests and their outcomes.

The Enterprise Service BUS (ESB) is a key component in SOA that offers transparent service endpoint location, routing, and translation for the Orchestrator, acting as a service proxy. In our approach, we assume that an ESB is in place, providing the mediation flow that executes the required integration of services for attending each request to the FHIR service.

Besides Orchestrator, AE Generator, and ESB, our SOA architecture includes FHIR services (i.e., loosely coupled software components used by healthcare organizations), some services related to access control (e.g., PDP, PIP), and the blockchain service.

3) BACKWARD ACCESS CONTROL SERVICE

The access control service integrates PEP, PAP, and PIP functions in our SOA (orange elements in Fig. 4). This service performs the enforcement of SoC's consent for data treatment. User-defined consents are translated to XACML access policies actionable by the PAP component of the access control service. Thus, access policies will contain rules that determine who can (and cannot) perform certain actions (i.e., REST operations) over FHIR resources [28], [29], [30].

Most control access mechanisms use a forward decision model (e.g., Fig. 2) by intercepting data access attempts and making a decision (i.e., grant or reject) before the actual data access takes place. However, due to FHIR Bundles, the actual resources that fulfil a request may be known only after the FHIR service has produced the response. Because of this, we opt for a backward decision model that works as follows (steps numbered in Fig. 4):

- The process orchestrator launches a service request (1) as part of a task in a business process in execution. The request is sent to the ESB, which redirects it to the corresponding FHIR service (2).
- After receiving the response as a Bundle including all resources that fulfil the request (3), the ESB will call the backward access control service (PEP component). The PEP asks for an authorization decision from the PDP for each resource in the response (4). The PDP, in turn, requests to the PAP all applicable policies for the operation (5) and makes a decision.
- Individual decisions are communicated to the PEP (6). It will censor unauthorized resources and strip them off from the service response.
- The modified service response (i.e., including only authorized resources) will then be routed to the Orchestrator (7).
- After the response, the PEP will also send the aforementioned data access-related information to the AE Generator residing in the Orchestrator (8).

4) GENERATION OF ACCESS LOGS AS FHIR AUDITEVENT RESOURCES

Access attempts and their corresponding outcomes should be registered as audit events along with all relevant information related. As mentioned above, the PEP notifies the AE Generator of any access attempt and outcome. When a business process completes its execution, the AE Generator aggregates in several *AuditEvent* resources all access attempts, requested resources, and the decisions made during the process and sends them to the blockchain (Fig. 4, step 9). Each *AuditEvent* will cover access attempts for a specific pair action-outcome (e.g., READ-granted).

TABLE 2. Simplified schema of the FHIR AuditEvent resource.

AuditEvent resource			
Resource Type	AuditEvent		
Id	Permitted-access-sample34		
Action	R*		
Recorded	2021-06-20T23:42:24Z		
Outcome	0		
Agent	Role: Grantor	type	Authorization server
		who	PDP-3425
	Role: Grantee	type	Healthcare provider
		who	Practitioner-77535
		purpose of use	Treatment
Source	type		BPM engine
	observer		jBPM
Entity	Role: Patient		Patient-9656
	References to resources accessed		

* READ action

Table 2 shows an example of an *AuditEvent* resource instance covering accesses with action Read and outcome success (codified as 0). It identifies involved agents (SoC,

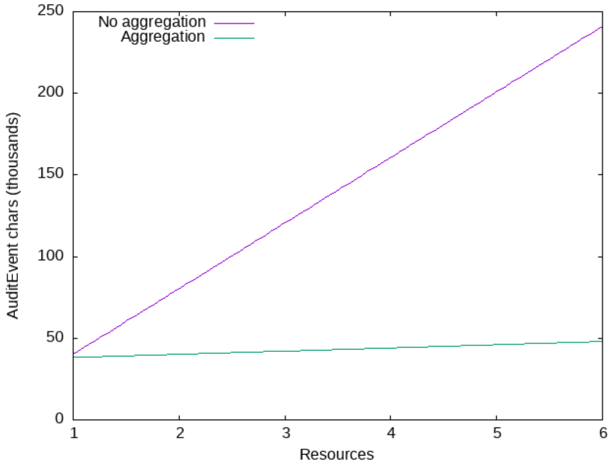


FIGURE 6. Size of AuditEvent resources with and without aggregation of access logs.

authorization service, data requester, interceptor PEP), timestamp, and entities/resources requested. Using a standardized format (i.e., FHIR) for storing audit events guarantees the interoperability and mutual understanding between healthcare organizations and facilitates the development of open solutions for auditing tasks performed by supervisory authorities or SoCs.

Although access logs could be generated by the access control service (i.e., in the context of request) and directly sent to the blockchain, performing it in the context of the business process incorporates some benefits:

1. An audit event generated from the business process viewpoint may aggregate numerous FHIR interactions during a process execution (e.g., a patient discharge or an encounter). Thus, it is possible to audit not only individual access to data but to relate all activities (and resources) involved in the business process. Awareness of well-documented high-level business processes is advisable in a complex context such as healthcare.
2. The aggregation of multiple access logs reduces the size of *AuditEvents* sent to the blockchain since common fields of the resource are shared and codified only once. The longer the process, the bigger the synthetization of generated *AuditEvents* (since they will include more events). Fig. 6 compares *AuditEvents* size (in characters) with and without aggregation when the number of accessed resources (of average size) increases.

5) BLOCKCHAIN FOR THE SUPPORT OF AUDITING TASKS

The last component of our scheme is a permissioned and private blockchain, to which each healthcare organization (and supervisory authorities) will contribute with a peer (i.e., blockchain gateway). Peers will access the blockchain through a REST service in our SOA. We use two separate channels for *Consents* and *AuditEvents* in the blockchain system. More details of the blockchain system are given in Section IV-A.

The availability of *AuditEvents* (and *Consents* ruling them) will allow participating entities to be capable of auditing GDPR compliance through a service that is out of our scope. The main beneficiaries of this functionality will be supervisory authorities (named by EU state members under GDPR) that, through their own blockchain gateways, will directly access to *AuditEvents* and *Consents*, with no intervention from healthcare organizations.

Additionally, since *AuditEvents* store information about resources implied in a business process (including references to the Patient, Practitioners, Task…), it is possible to perform process-aware auditing with guarantees of immutability and tamper-proof provided by the blockchain. These characteristics also allow the traceability of access attempts and outcomes. Thus, identity theft and malicious insider within organizations can be detected by analyzing blockchain transactions.

There are several threats to using blockchain technologies in healthcare [31]. The main organizational threat is interoperability due to the lack of trust between parties, limited open standards, and the initial installation cost. The main technical threats arise from the scalability of blockchain, which is related to the trade-off between the transaction volume and the computer power required to handle the transaction. In our case, using open standards and a permissioned and private blockchain addresses these issues, as proper security configuration can be applied on the server side (including access-list control), and the deployment cost is reduced.

IV. PROOF OF CONCEPT AND EXPERIMENTAL RESULTS

We have developed a prototype to assess the feasibility of our approach and show the configuration of a blockchain in our context. We also have performed a set of experiments to analyze scalability. Although a BPM-driven SOA has been implemented, each organization will deploy a specific architecture in practice. Thus, the results from experimentation with SOA would be meaningless for actual practice and are not explored here. On the contrary, the components shared by all organizations (i.e., blockchain platform) can be leveraged into real deployment, so its performance is evaluated via experiments.

A. IMPLEMENTATION

1) HEALTHCARE SOA ARCHITECTURE

Our demonstrator required deploying an access control service based on BPM and SOA for organizations participating in the blockchain. WSO2 is an open-source provider that integrates existing access control solutions and provides interoperable SOA components [32]. The WSO2 SOA middleware is Carbon. We used WSO2 API Manager [33] and Identity Server (IS) [34] for the access control service. The API Manager is used to publish the FHIR API REST and to host the PEP as a customized mediation sequence that manages the flow of FHIR interactions. It is deployed in its embedded micro integrator or API Gateway. The IS includes

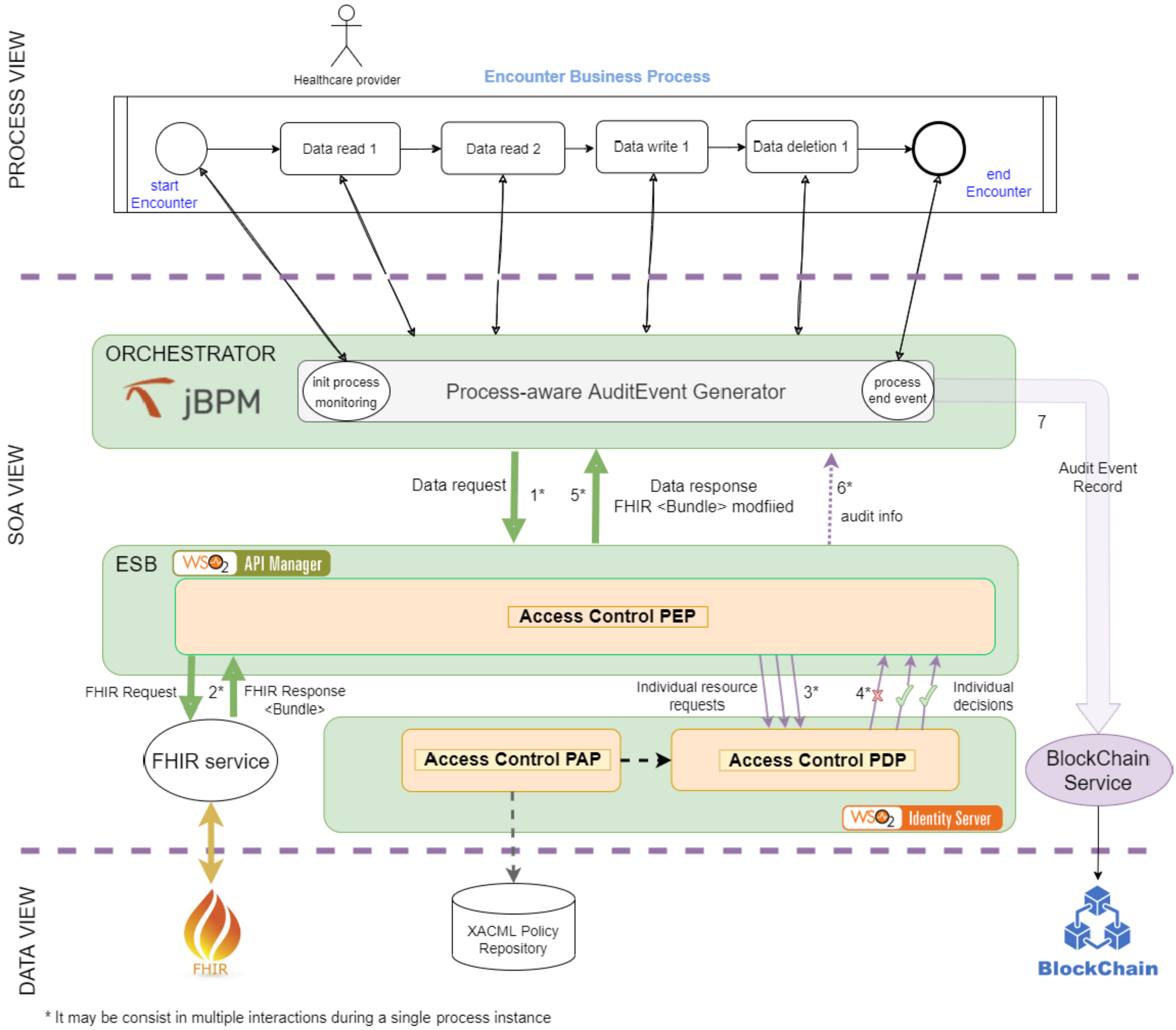


FIGURE 7. Architecture of the backward authorization service in the healthcare SOA.

PAP and PDP components with standard interfaces that conform to SAML and XACML. The WSO2 API Manager and IS have been instantiated in a virtual machine (VM) each. Another VM has been deployed to execute the backend FHIR server. Although the proof-of-concept is built upon WSO2, our architecture could be implemented with other tools thanks to the adoption of security and health information exchange standards.

jBPM [35] has been used as a business process workflow engine and it has been deployed on a dedicated VM, including communication with PEP (via REST interfaces for service request/response and JMS as a publish/subscribe mechanism for audit info exchange) and the creation of *AuditEvents*.

Fig. 7 shows the architecture of the implementation setup.

For the experimental analysis, we focus on a generic clinical process in a healthcare organization, that is, an encounter between a SoC and a healthcare provider with the purpose of providing healthcare services or assessing the health status of the SoC. During the execution of an instance of an encounter process, each FHIR interaction (e.g., requesting observations related to the SoC) is sent to the corresponding FHIR service endpoint that triggers the PEP function (1). Following the proposed backward model, the PEP sends the interaction to the FHIR backend server, in order to get the response containing one or several resources. The FHIR server sends back a FHIR response (2). If the response is a Bundle (i.e., a container of all instances that fulfil the request), the PEP builds an XACML access decision query for each instance in the Bundle, sending all to the PDP (3). The PDP gathers from the PAP the applicable policies for each requested resource and returns a decision response to the PEP for each requested resource (4). Finally, the PEP will delete all resources denied by the PDP from the Bundle. Thus, the mediation sequence censures the FHIR response before being returned to the requester (5).

Since all access control decisions must be included in *AuditEvents*, the PEP sends any relevant and auditable access information to the orchestrator (6). Before the process instance ends, the AE Generator will build the corresponding *AuditEvents*.

In order to increase efficiency, the jBPM engine will aggregate all the events that occurred during an Encounter process into several *AuditEvents*. Objects accessed with the same operation and outcome are aggregated in the same *AuditEvent*. Thus, five operations (create, read, update, delete and execute) and two outcomes (permit/deny) resulting in a maximum of 10 *AuditEvent* instances created after the encounter (worst scenario) and sent to the blockchain gateway for persistence (7). In Table 3, the size of the *AuditEvent* instances is estimated according to the IHE Basic Audit Log Patterns (BALP) [36].

TABLE 3. Estimated size (in characters) of AuditEvent instances created after one encounter.

		Average size (# of characters)
Fixed content		
AuditEvent	Permit	3331
instance	Deny	3921
Variable content		
1 FHIR resource		391

Let Ni be the number of FHIR resources during the encounter i. Considering the best case, in which access to all requested FHIR resources has been permitted and just one REST action (e.g., read), the number of characters to insert in the blockchain generated during the encounter would be 3331 (fixed part of permit) + Ni*391. On the contrary, the worst case will consist of using the five types of REST actions (create, read, update, delete, and execute), resulting in both permitted and denied FHIR resources for each action. In this case, the number of characters to insert in the blockchain would be (3331+3921) * 5 + Ni * 391 = 36260 + Ni * 391.

2) BLOCKCHAIN

The blockchain network has been developed and deployed using HyperLedger [24]. The Hyperledger Fabric supports arbitrary smart contracts or application-specific trust models to validate transactions. The Hyperledger Fabric transaction workflow involves four steps:

1. Endorsement phase – A client application sends a transaction proposal (i.e., a request to invoke a chaincode function with certain input parameters with the intent of reading and/or updating the ledger) to one or more endorsing peers. These execute the chaincode locally and send back the proposal response (signed by the endorsing peer) to the client application. No updates are made to the ledger at this point.
2. Ordering Phase –The application creates and sends a ‘‘transaction message’’ to the ordering service. This message will contain the read/write sets, the endorsing peers’ signatures, and the communication channel ID. The ordering service receives transactions from all channels and creates signed blocks of transactions, which are delivered to all peers on the channel.
3. Validation Phase – Peers on a channel validate the transactions within the block received to ensure that (a) the endorsement policy is fulfilled and (b) that there have been no changes to the ledger state for read set variables since the read set was generated by the transaction execution.
4. Ledger Update Phase – each peer appends the block to the channel’s chain, and for each valid transaction, the write sets are committed to the current state database. The client application is notified.

The previous transaction workflow is called *consensus,* since every peer has agreed on the order and the content of transactions.

Our test Fabric network consists of 4 organizations connected through two independent communication channels (for *Consents* and *AuditEvents* chaincodes). One of the organizations is meant to provide two Orderer nodes for redundancy. The rest, represent three different hospitals that may contribute to both chaincodes, so each of them implements two endorsing peers that are joined to the *Consents* and *AuditEvents* channels, respectively, also having its respective chaincode installed. Finally, each organization provides its own Certificate Authority node to handle the cryptographic material of the network (Fig. 8).

In our implementation, we used CouchDB for ledger storage, Fabric-ca for certificate authorities, and the endorsement policy has been set so it requires two out of three hospital organizations to acknowledge in the endorsing phase. Chaincodes have been defined as Smart Contracts, hence defining the interaction rules between users and both channels. An excellent guide for tuning the configuration parameters of the blockchain can be found in [21]. Each peer has been deployed using docker containers as shown in Table 4. To do so, we parted from the official Hyperledger Fabric samples repository, modifying *test-network* and *asset-transfer-basic* modules to implement the scenario described above.

The hardware platform used in our implementation was an Intel ®i5-8400 @ 2.8GHz with 16 GB of RAM and 6 cores. We run Manjaro Linux 21.2.6 as operating system and Hyperledger v2.4 (last stable) nodes using Dockers v20.10 with Docker Composer v1.25.0.

B. EXPERIMENTS

To evaluate the capacity of the system, three healthcare organizations were simulated during 12 working hours. In the simulation, each healthcare organization attends a uniformly

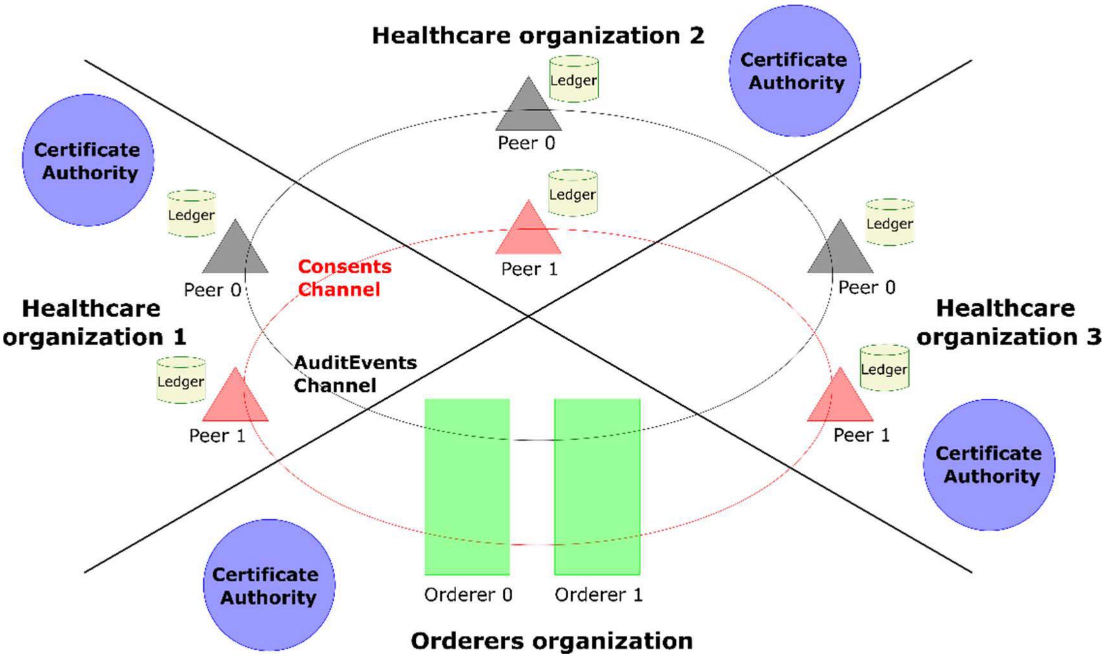


FIGURE 8. Blockchain implementation.

TABLE 4. HyperLedger dockers deployment in our test fabric network.

3 healthcare organizations with 2 channels scenario	
Number of Dockers	Type
6	Endorsing Peers nodes (one by channel and organization)
2	Service Orderer nodes
4	Certification Nodes (one by organization)
6	CouchDB instances (one by peer)

distributed number of encounters per hour (i.e., interarrival times are deterministic). Fig. 9 illustrates the sequence diagram of events simulated for each encounter, representing the following entities: SoC, practitioner, Authorization Service (i.e., which includes SoCs' consents app, healthcare organization access control software –PEP, PAP, and PDP-, and the XACML Policy Repository), blockchain system with both *Consents* and *AuditEvents* chaincodes, and FHIR server.

At each encounter, SoC sets four consents by using the consent management service (embedded into the Authorization Service in Fig. 9). As a result, it queries four sequential transactions into the *Consents* chaincode. Each consent refers to a single resource, which accounts for a particular piece of personal health data. In order to test four different action-outcome pairs, consents are set so they (a) allow read of resource 1, (b) deny read of resource 2, (c) allow update of resource 3, and (d) deny update of resource 4. Then, once all consents are committed into the blockchain, an encounter between the SoC and a practitioner begins. The practitioner asks the authorization Service (via the orchestrator) to read resources 1 and 2, and to update resources 3 and 4. These actions return the following action-outcome pairs (respectively): read-grant, read-deny, update-grant, and update-deny. Finally, the audit info of each outcome is committed to the *AuditEvents* chaincode as four sequential transactions.

In summary, each encounter triggers four *Consents* transactions (120B on average) and four *AuditEvent* transactions (4017B on average). Although *Consents* transactions could be aggregated into a single proposal, we have chosen to avoid aggregation to represent the worst-case scenario, hence speeding up the saturation condition. Fig. 10 shows the transaction times of the *Consents* and *AuditEvents* chaincode (as a rolling average of 5 minutes) while increasing the SoC arrival rate from 1000 to 1700 SoC per hour per healthcare organization. The measured transaction times are highlighted in green in Fig. 10. Triangle markers at the end of any curve represent the blockchain network failure (connection, endorsing or committing timeouts). The CPU and memory resources were not exhausted during experiments, so we will assume that the docker environment was not a bottleneck.

Finally, an auditing application has been implemented to test whether the audit procedure could be carried out in a reasonable time. This application checks if each consent granted in the *AuditEvents* blockchain is legit by exploring the *Consents* ledger history. For the 1000 SoCs/hour experiment, it can generate a daily report in less than 1.5 minutes. Note that the auditing application does not require transactions since it queries a peer ledger locally, so there is no endorsing or committing delays.

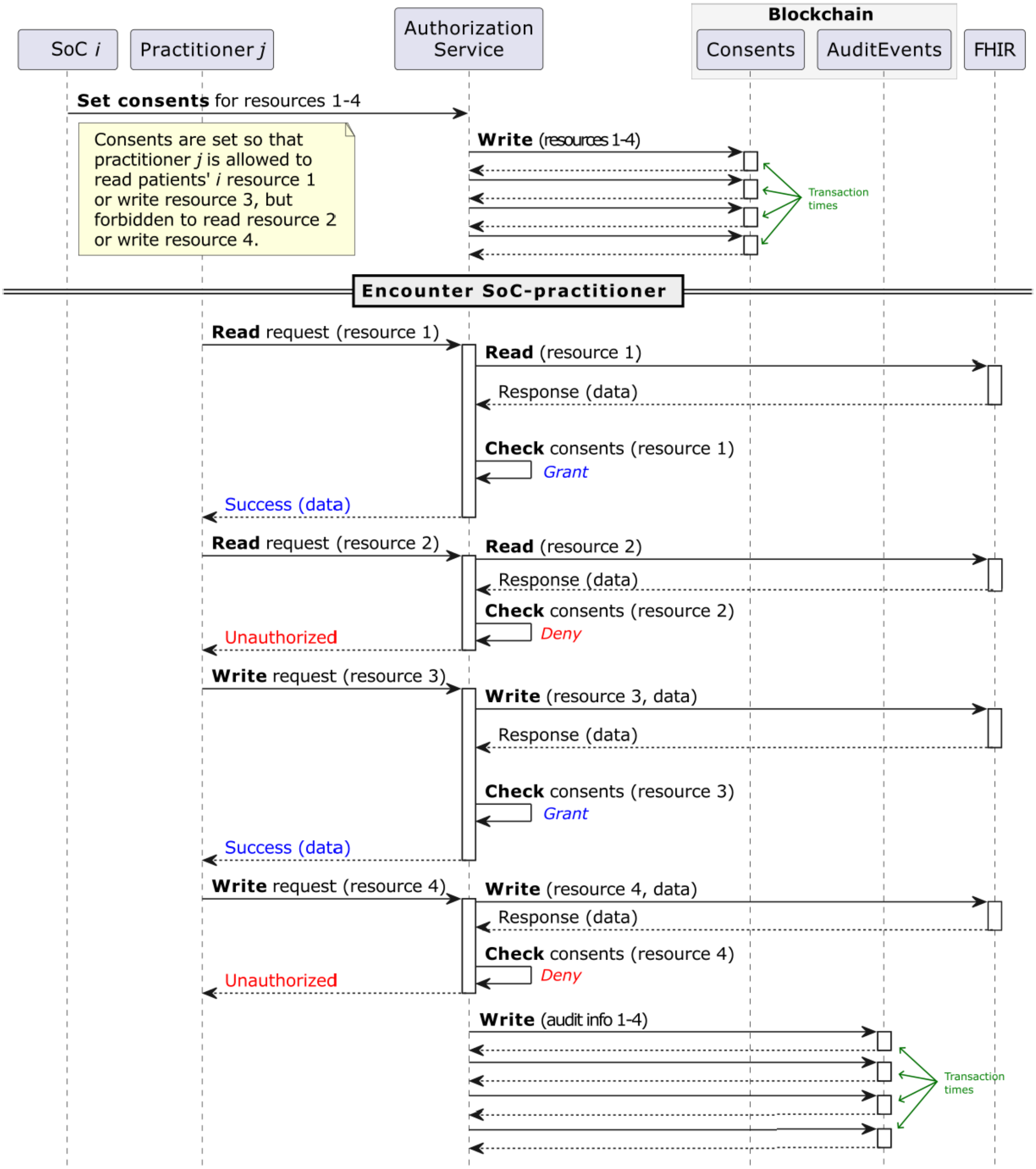

SoC *i*
Practitioner *j*
Authorization Service
Blockchain
Consents
AuditEvents
FHIR
Set consents for resources 1-4
Consents are set so that practitioner *j* is allowed to read patients' *i* resource 1 or write resource 3, but forbidden to read resource 2 or write resource 4.
Write (resources 1-4)
Transaction times
Encounter SoC-practitioner
Read request (resource 1)
Read (resource 1)
Response (data)
Check consents (resource 1)
Grant
Success (data)
Read request (resource 2)
Read (resource 2)
Response (data)
Check consents (resource 2)
Deny
Unauthorized
Write request (resource 3)
Write (resource 3, data)
Response (data)
Check consents (resource 3)
Grant
Success (data)
Write request (resource 4)
Write (resource 4, data)
Response (data)
Check consents (resource 4)
Deny
Unauthorized
Write (audit info 1-4)
Transaction times

FIGURE 9. Sequence diagram of events simulated for one encounter.

V. DISCUSSION

Some features of our approach and results deserve to be discussed. First, the backward model used is a consequence of the complexity of responses to FHIR interactions, but other alternatives may be proposed. The first alternative is to use an advanced resource PIP that stores information about the resources and their relations. In such an approach, PEP would send to PDP the original FHIR interaction (following the traditional forward model), and PDP would ask PIP which resources are involved in the interaction, making the decision afterward. However, this model does not exempt PEP from performing the original interaction, since it will receive a set of decisions from PDP that it needs to interpret in order to send the response to the requester. As a result, this alternative does not mean a significant advantage over our approach. Indeed, the backward model delegates the process logic to PEP, which allows the use of a PDP completely agnostic to FHIR and eliminates the need to develop a customized PIP.

Another alternative would be adopting the XACML Multiple Decision Profile [37] to avoid manifold queries to the PDP. This profile presents a schema for PEP to formulate multiple access control queries in a single XACML decision

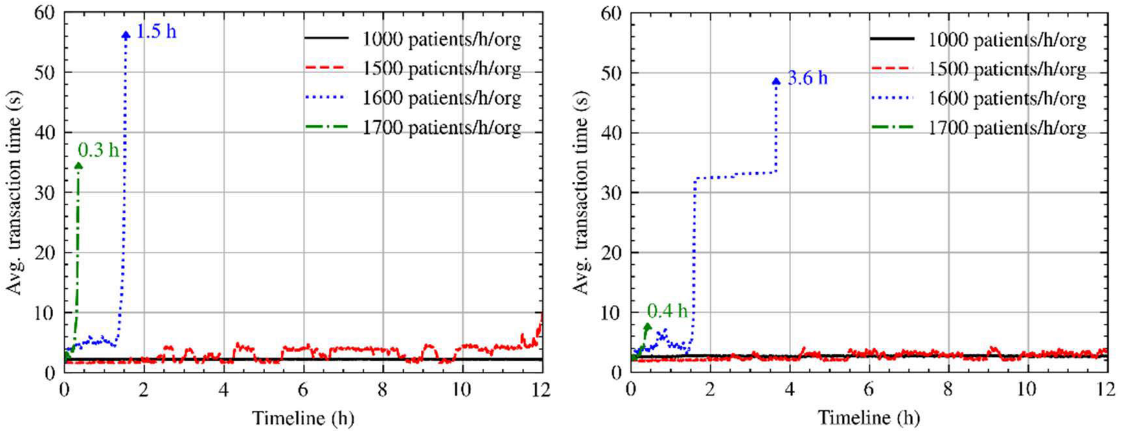


FIGURE 10. Average transaction delay for the Consents chaincode (left) and the AuditEvents chaincode (right).

request in an efficient way, and accordingly obtain a set of responses from PDP, one for each individual query. Currently, WSO2 supports this profile, but only for one query in each request to the PDP. Therefore, a new mediation sequence is required, which would modify the decision algorithm. The implementation of this profile is scarce in other tools different from WSO2, whereas our proposed model, adjusted to the basics of XACML, could be implemented with a wide variety of solutions compliant with the standard.

Regarding the blockchain, the registration of access attempts could be performed individually instead of grouping all events occurring during the same process. Nonetheless, the individual registration of events presents several disadvantages. For instance, an interaction may be repeatedly executed during a process, obtaining the same response each time. Therefore, instead of persisting a single event for all interactions, a set of events with the same information would be stored, with no relevance for auditing but increasing the size (or number) of blocks in the chain. Additionally, if events were captured at the interaction level, metadata provided by context would not be available, and it would be needed to add it later, which is harder than managing an event at the process level, as we proposed. Furthermore, our approach optimizes the amount of data sent to the blockchain and, by following the standardized format of FHIR, facilitates the traceability of audit events.

An important data privacy requirement defined in the GDPR is the 'right to be forgotten', that is, an individual can request organizations to completely delete her personal data. The immutability of blockchain results in blocks that cannot be erased, which could hamper the fulfilment of this requirement. However, in our approach, the blockchain includes the SoC id in *Consents* and *AuditEvents*, but pseudonymized. If an individual requires to be forgotten, organizations will erase her data and the link between her id and the identifiable person. Although the id remains on the blockchain, no organization will be able to identify a SoC via that deprecated id.

Our experiments show that with a modest hardware platform for the blockchain implementation, we carried out up to 1500 SoCs/h per organization (4500 SoCs/hour) in the worst-case scenario with no aggregation at the *Consents* proposals, which should suffice for most cases. Notice how a higher encounter rate reaches saturation condition in a few hours, thus resulting in the blockchain network failure. Keep in mind that our consensus policy does not imply any proof of work, as could happen with other blockchain networks like Bitcoin or Ethereum. On the other hand, communications between different organization servers through a network would also impact performance in real scenarios. If a higher number of concurrent EHR users is needed, *Consent* transactions could be grouped before being inserted into the blockchain, hence reducing the number of transactions per second at the cost of reducing interactivity with the blockchain since the enqueuing would add extra delays.

Finally, we must note that this is a preliminary work and the experimental results do not validate the whole approach, including SOA, but only the blockchain platform. Nevertheless, the results support the application of blockchain for consent and audit data exchange.

VI. CONCLUSION

This work presents an approach for auditing access to health resources based on individual consent by combining SOA, BPM, and blockchain technologies for healthcare. Audit events and consents are stored in a chain and shared by all involved parties. Through blockchain, supervisory authorities (or SoCs themselves) can audit access to protected resources without intervention from healthcare organizations, which guarantees nonrepudiation and tamper-proof information. This direct auditing method eases third-party assessment of GDPR compliance performed by organizations storing personal health data. In addition, a fine-grained backward access control decision model is proposed to address the requirements of complex FHIR interactions. Furthermore, the adoption of standards enhances the interoperability and openness required for distributed environments. Finally, the experimental results validate the feasibility of using a blockchain-supported architecture for consent management, access control, and auditing in the health domain.

Some limitations of this work may be identified. First, the results depend strongly on the healthcare business process, so they should be taken merely as an orientation. Second, a blockchain has been implemented for a single configuration tested in Docker containers (same computer), which limits the generalization of results.

Among future steps of this research, we consider: adjusting the number of FHIR interactions and resources involved in processes by requesting real data; implementing a blockchain distributed on different computers to assess network influence; testing different blockchain configurations such as endorsement policies or ledger database; advancing in the backward access control model by including resources PIP that store resource attributes and support the decision making but, at the same time, stay FHIR-agnostic.

REFERENCES

[1] *EU General Data Protection Regulation*, EU Regulation 2016/679, Eur. Union, Europe, 2016.

[2] S. Daoudagh, E. Marchetti, V. Savarino, R. Di Bernardo, and M. Alessi, "How to improve the GDPR compliance through consent management and access control," in *Proc. 7th Int. Conf. Inf. Syst. Secur. Privacy*, 2021, pp. 534–541.

[3] A. Hasselgren, K. Kralevska, D. Gligoroski, S. A. Pedersen, and A. Faxvaag, "Blockchain in healthcare and health sciences—A scoping review," *Int. J. Med. Informat.*, vol. 134, Feb. 2020, Art. no. 104040, doi: 10.1016/j.ijmedinf.2019.104040.

[4] M. M. Merlec, Y. K. Lee, S.-P. Hong, and H. P. In, "A smart contract-based dynamic consent management system for personal data usage under GDPR," *Sensors*, vol. 21, no. 23, p. 7994, Nov. 2021, doi: 10.3390/s21237994.

[5] K. Rantos, G. Drosatos, A. Kritsas, C. Ilioudis, A. Papanikolaou, and A. P. Filippidis, "A blockchain-based platform for consent management of personal data processing in the IoT ecosystem," *Secur. Commun. Netw.*, vol. 2019, pp. 1–15, Oct. 2019, doi: 10.1155/2019/1431578.

[6] D. Tith, J.-S. Lee, H. Suzuki, W. M. A. B. Wijesundara, N. Taira, T. Obi, and N. Ohyama, "Patient consent management by a purpose-based consent model for electronic health record based on blockchain technology," *Healthcare Informat. Res.*, vol. 26, no. 4, pp. 265–273, Oct. 2020, doi: 10.4258/hir.2020.26.4.265.

[7] T. Rupasinghe, F. Burstein, and C. Rudolph, "Blockchain based dynamic patient consent: A privacy-preserving data acquisition architecture for clinical data analytics," in *Proc. ICIS*, Munich, Germany, 2019, pp. 1–9.

[8] V. Jaiman and V. Urovi, "A consent model for blockchain-based health data sharing platforms," *IEEE Access*, vol. 8, pp. 143734–143745, 2020, doi: 10.1109/ACCESS.2020.3014565.

[9] C. C. Agbo and Q. H. Mahmoud, "Design and implementation of a blockchain-based E-health consent management framework," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Toronto, ON, Canada, Oct. 2020, pp. 812–817.

[10] G. Albanese, J.-P. Calbimonte, M. Schumacher, and D. Calvaresi, "Dynamic consent management for clinical trials via private blockchain technology," *J. Ambient Intell. Humanized Comput.*, vol. 11, no. 11, pp. 4909–4926, Nov. 2020, doi: 10.1007/s12652-020-01761-1.

[11] P. V. Kakarlapudi and Q. H. Mahmoud, "Design and development of a blockchain-based system for private data management," *Electronics*, vol. 10, no. 24, p. 3131, Dec. 2021, doi: 10.3390/electronics10243131.

[12] T. M. Kim, S.-J. Lee, D.-J. Chang, J. Koo, T. Kim, K.-H. Yoon, and I.-Y. Choi, "DynamiChain: Development of medical blockchain ecosystem based on dynamic consent system," *Appl. Sci.*, vol. 11, no. 4, p. 1612, Feb. 2021, doi: 10.3390/app11041612.

[13] *Information Technology—Open Systems Interconnection—Security Frameworks for Open Systems: Access Control Framework*, ITU-T Standard X.812, 1995.

[14] *eXtensible Access Control Markup Language (XACML)*, OASIS Standard TC R3.0, 2013.

[15] *Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML)*, OASIS Standard TC R2.0, 2005.

[16] V. C. Hu, D. R. Kuhn, and D. F. Ferraiolo, "Attribute-based access control," *Computer*, vol. 48, no. 2, pp. 85–88, Feb. 2015, doi: 10.1109/MC.2015.33.

[17] K. Albulayhi, A. Abuhussein, F. Alsubaei, and F. T. Sheldon, "Fine-grained access control in the era of cloud computing: An analytical review," in *Proc. 10th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Las Vegas, Nevada, Jan. 2020, pp. 748–755.

[18] *Fast Healthcare Information Resources (FHIR)*. Accessed: Feb. 1, 2023. [Online]. Available: https://www.hl7.org/fhir/

[19] E. Helm, O. Krauss, A. Lin, A. Pointner, A. Schuler, and J. Kung, "Process mining on FHIR—An open standards-based process analytics approach for healthcare," in *Proc. Int. Conf. Process Mining*, 2020, pp. 343–355.

[20] M. J. M. Chowdhury, M. S. Ferdous, K. Biswas, N. Chowdhury, A. S. M. Kayes, M. Alazab, and P. Watters, "A comparative analysis of distributed ledger technology platforms," *IEEE Access*, vol. 7, pp. 167930–167943, 2019, doi: 10.1109/ACCESS.2019.2953729.

[21] P. Thakkar, S. Nathan, and B. Viswanathan, "Performance benchmarking and optimizing hyperledger fabric blockchain platform," in *Proc. IEEE 26th Int. Symp. Model., Anal., Simul. Comput. Telecommun. Syst. (MASCOTS)*, Milwaukee, Wisconsin, Sep. 2018, pp. 264–276.

[22] S. Nakamoto. (2008). *A Peer-To-Peer Electronic Cash System*. Accessed: Feb. 1, 2023. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[23] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, Apr. 2017.

[24] *Open Source Blockchain Technologies*. Accessed: Feb. 1, 2023. [Online]. Available: https://www.hyperledger.org/

[25] M. Chinosi and A. Trombetta, "BPMN: An introduction to the standard," *Comput. Standards Interfaces.*, vol. 34, no. 1, pp. 124–134, Jan. 2012, doi: 10.1016/j.csi.2011.06.002.

[26] C. Ouyang, M. Dumas, A. Ter Hofstede, and W. Van Der Aalst, "From BPMN process models to BPEL web services," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Chicago, Illinois, Sep. 2006, pp. 285–292.

[27] A. Arsanjani, "Service-oriented modeling and architecture," *IBM Developer Works*, vol. 1, p. 15, Nov. 2004.

[28] Y. K. Rivera Sánchez, S. A. Demurjian, and M. S. Baihan, "A service-based RBAC & MAC approach incorporated into the FHIR standard," *Digit. Commun. Netw.*, vol. 5, no. 4, pp. 214–225, Nov. 2019, doi: 10.1016/j.dcan.2019.10.004.

[29] S. Pal, M. Hitchens, V. Varadharajan, and T. Rabehaja, "Policy-based access control for constrained healthcare resources in the context of the Internet of Things," *J. Netw. Comput. Appl.*, vol. 139, pp. 57–74, Aug. 2019, doi: 10.1016/j.jnca.2019.04.013.

[30] K. Riad and J. Cheng, "Adaptive XACML access policies for heterogeneous distributed IoT environments," *Inf. Sci.*, vol. 548, pp. 135–152, Feb. 2021, doi: 10.1016/j.ins.2020.09.051.

[31] I. Abu-elezz, A. Hassan, A. Nazeemudeen, M. Househ, and A. Abd-Alrazaq, "The benefits and threats of blockchain technology in healthcare: A scoping review," *Int. J. Med. Informat.*, vol. 142, Oct. 2020, Art. no. 104246, doi: 10.1016/j.ijmedinf.2020.104246.

[32] *WSO2*. Accessed: Feb. 1, 2023. [Online]. Available: https://wso2.com

[33] *WSO2 API Manager*. Accessed: Feb. 1, 2023. [Online]. Available: https://apim.docs.wso2.com/en/latest/

[34] *WSO2 Identity Server*. Accessed: Feb. 1, 2023. [Online]. Available: https://is.docs.wso2.com/en/latest/

[35] *jBPM*. Accessed: Feb. 1, 2023. [Online]. Available: https://www.jbpm.org/

[36] (2022). *Basic Audit Log Patterns (BALP)*. Accessed: Feb. 1, 2023. [Online]. Available: https://profiles.ihe.net/ITI/BALP/index.html

[37] *XACML V3.0 Multiple Decision Profile 1.0*, OASIS Standard, OASIS, USA, 2014.

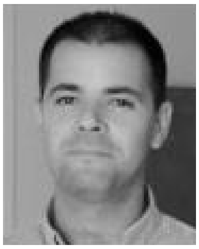
ISABEL ROMÁN-MARTÍNEZ received the M.S. and Ph.D. degrees in telecommunication engineering from the University of Seville, Spain, in 1999 and 2006, respectively. She currently works as an Associate Professor and a Researcher with the Technical School of Engineering, Seville. Her research interests include systems interoperability in the healthcare domain, including topics, such as middleware, architectural paradigms, and normalization.

JORGE CALVILLO-ARBIZU received the M.S. and Ph.D. degrees in telecommunications engineering from the University of Seville, in 2007 and 2013, respectively. He currently works as an Assistant Professor with the Department of Telematics Engineering and belongs to the Biomedical Engineering Group, University of Seville. His research interests include security (mainly, distributed access control on the health domain), interoperability (standard-based exchange of health information and distributed system architectures), and ICT-based patient empowerment.

VICENTE J. MAYOR-GALLEGO received the M.S. and Ph.D. degrees in telecommunication engineering from the University of Seville, in 2017 and 2020, respectively. In the past, he has worked for a year as a Research and Development Project Manager at Wellness TechGroup, Spain. He is currently an Associate Professor with the Department of Telematics Engineering, University of Seville. His research interests include the areas of networking, voice over IP (VoIP), the quality of service, wireless networks, UAV-assisted wireless coverage, and cybersecurity.

GERMÁN MADINABEITIA-LUQUE received the M.S. and Ph.D. degrees in telecommunication engineering from the Universidad Politecnica de Madrid, in 1986 and 2004, respectively. In the past, he was worked for ten years as a Product Engineer in the industry. He is currently an Assistant Professor with the Department of Telematics Engineering, University of Seville. His research interests include the areas of networking, the Internet of Things, cybersecurity, and traffic engineering.

ANTONIO J. ESTEPA-ALONSO received the M.S. and Ph.D. degrees in telecommunication engineering from the University of Seville, in 1998 and 2004, respectively. In 2004, he was also a Visitor with the Department of Electrical Engineering and Computer Science, University of Minnesota, USA. He is currently an Associate Professor with the Department of Telematics Engineering, University of Seville. From 1998 to 2000, he was a Software and Network Engineer with a software development company. He has authored or coauthored several journals or conferences papers. His research interests include the areas of telecommunication networks, with particular emphasis in networking protocols, wireless networks, and cybersecurity.



RAFAEL M. ESTEPA-ALONSO received the M.S. and Ph.D. degrees in telecommunication engineering from the University of Seville, in 1998 and 2002, respectively. He is currently an Associate Professor with the Department of Telematics Engineering, University of Seville. In the past, he was worked for two years as a Product Engineer with Alcatel, Spain. He has also been a Visitor with the Department of Applied Mathematics, Instituto Superior Tecnico (IST), Lisbon, and the Dublin Institute of Technology (DIT). His research interests include the areas of networking, voice over IP (VoIP) quality of service, wireless networks, unmanned aerial vehicles (UAVs), and cybersecurity.