# D3.8

# Specification of consent management and decentralized authorization mechanisms for HR Exchange - V2


ABSTRACT

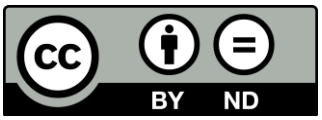
This deliverable provides the second and final version of the specification of consent management and decentralized authorization mechanisms for health records in InteropEHRate. This document also provides a detailed technical background for consent management, authorization and blockchain mechanisms, including other relevant projects, which is a necessary step to move forward. The deliverable includes the consent management and authorization aspects of all the involved architecture components (e.g., S-EHR App, HCP Web App, S-EHR Cloud, Central Node and Reference Research Center), protocols (e.g., D2D, R2D Access, R2D Backup, R2D Emergency, RDS), and scenarios (e.g., Medical Visit, Emergency and Research) for HR exchange.


| | |
|---|---|
| **Delivery Date** | September 9th, 2021 |
| **Work Package** | WP3 |
| **Task** | T3.3 |
| **Dissemination Level** | Public |
| **Type of Deliverable** | Report |
| **Lead partner** | UBIT |




InteropEHRate project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826106.

This document has been produced in the context of the InteropEHRate Project which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826106. All information provided in this document is provided "as is" and no guarantee or warranty is given that the information is fit for any particular purpose.



## CONTRIBUTORS

| | Name | Partner |
|---|---|---|
| **Contributors** | Sofianna Menesidou, Entso Veliou, Menelaso Giannopoulos, Thanassis Giannetsos | UBIT |
| | Chrysostomos Symboulidis | BYTE |
| | Thanos Kiourtis | UPRC |
| **Reviewers** | Gabor Bella | UNITN |
| **Reviewers** | Francesco Torelli | ENG |

## LOGTABLE

| Version | Date | Change | Author | Partner |
|---|---|---|---|---|
| 0.1 | 12-05-21 | ToC and cleaning from the v1 | Sofianna Menesidou | UBIT |
| 0.2 | 28-06-21 | Introduction | Sofianna Menesidou | UBIT |
| 0.3 | 01-07-21 | Section 3 | Sofianna Menesidou | UBIT |
| 0.4 | 08-07-21 | Section 3 | Sofianna Menesidou | UBIT |
| 0.5 | 13-07-21 | Section 3 | Thanos Kiourtis | UPRC |
| 0.6 | 14-07-21 | Section 3 | Chrysostomos Symboulidis | BYTE |
| 0.7 | 20-07-21 | Section 3 | Sofianna Menesidou | UBIT |
| 0.8 | 21-07-21 | Section 3 | Sofianna Menesidou | UBIT |
| 0.9 | 23-07-21 | Section 3 | Sofianna Menesidou | UBIT |
| 1.0 | 04-08-21 | Section 3 | Sofianna Menesidou | UBIT |
| 1.1 | 12-08-21 | Section 3, Section 5 | Sofianna Menesidou | UBIT |
| 1.2 | 23-08-21 | Section 3, Section 5 | Sofianna Menesidou | UBIT |
| 1.3 | 26-08-21 | Section 3, Section 5 | Sofianna Menesidou | UBIT |
| 1.4 | 08-09-21 | Updates after the internal review | Sofianna Menesidou | UBIT |
| Vfinal | 09-09-21 | Quality check for submission | Laura Pucci | ENG |

ACRONYMS

| Acronym | Description |
|---|---|
| ABAC | Attribute Based Access Control |
| APPC | Advanced Patient Privacy Consents |
| BPPC | Basic Patient Privacy Consents |
| CM | Consent Management |
| D2D | Device-to-Device |
| DAC | Discretionary Access Control |
| HER | Electronic Health Records |
| EPP | Event Processing Point |
| GDPR | General Data Protection Regulation |
| HCP | Health Care Professional |
| HR | Health Record |
| IDM | Identity Management |
| MAC | Mandatory Access Control |
| MD2DI | Mobile Device-to-Device Interface |
| NGAC | Next Generation Access Control |
| PAP | Policy Administration Point |
| PAP | Policy Access Point |
| PBFT | Practical Byzantine Fault Tolerance |
| PDP | Policy Decision Point |
| PEP | Policy Enforcement Point |
| PHI | Protected Health Information |
| PHR | Patient Health Records |
| PIP | Policy Information Point |

| | |
|---|---|
| PoW | Proof-of-Work |
| PoS | Proof of Stake |
| R2D | Remote-to-Device |
| R2R | Remote-to-Research |
| R2DI | Remote-to-Device Interface |
| RAP | Resource Access Points |
| RBAC | Role-Based Access Control |
| RSI | Research Interface |
| S-HER | Smart Electronic Health Record |
| SDK | Software Development Kit |
| XACML | eXtensible Access Control Markup Language |

TABLE OF CONTENT

1 INTRODUCTION ..... 1
1.1 Scope of the document ..... 1
1.2 Intended audience ..... 1
1.3 Structure of the document ..... 1
1.4 Updates with respect to previous version (if any) ..... 1
2 TECHNICAL BACKGROUND ..... 3
2.1 Consent Management ..... 3
2.1.1 Existing Standards for structural representation of consents ..... 6
2.2 Decentralized Authorization ..... 7
2.2.1 Existing ABAC Standards ..... 9
2.3 Blockchain and Distributed Ledger Technologies ..... 11
2.3.1 Blockchain Usage in Healthcare ..... 12
2.3.2 Blockchain Basic Elements, Types and Consensus Algorithms ..... 13
2.3.3 Smart Contracts ..... 15
2.3.4 Popular Blockchain Platforms ..... 16
2.4 Relation with other research projects ..... 17
2.4.1 LETITFLOW - Active Distributed Workflow System For Elderly (AAL Programme) ..... 18
2.4.2 SPHINX – A Universal Cyber Security Toolkit for Health-Care Industry (H2020) ..... 19
2.4.3 Blockchain usage in healthcare projects ..... 19
3 INTEROPEHRATE CONSENT MANAGEMENT AND DECENTRALIZED AUTHORIZATION ..... 21
3.1 D2D Security Architecture and Models ..... 22
3.2 D2D Security APIs ..... 25
3.2.1 S-EHR App Security APIs ..... 25
3.2.2. HCP App Security APIs ..... 26
3.3 R2D Access Security Architecture and Models ..... 26
3.4 R2D Access Security APIs ..... 28
3.5 R2D Backup and Emergency Security Architecture and Models ..... 28
3.6 R2D Backup and Emergency Security APIs ..... 34
3.6.1 S-EHR Cloud Security APIs ..... 34
3.6.2 Health Organisation Security APIs ..... 40
3.7 RDS Security Architecture and Models ..... 41

3.8 RDS Security APIs ..... 43
3.8.1 RRC App Security APIs ..... 43
3.8.2 Health Organisation (R2R Access) Security APIs ..... 46
3.9 RDS Blockchain Optional Service and APIs ..... 47
4 ALIGNMENT WITH STANDARDS AND GUIDELINES ..... 51
5 CONCLUSIONS AND NEXT STEPS ..... 52

## LIST OF FIGURES

Figure 1 - Consent Process Flow
Figure 2 - ABAC Indicative Information Flow
Figure 3 - XACML data-flow diagram
Figure 4 - NGAC data-flow diagram
Figure 5 – InteropEHRate protocols
Figure 6 – D2D crypto model
Figure 7 – D2D sequence diagram
Figure 8 – R2D Access crypto-model
Figure 9 – R2D Access sequence diagram
Figure 10 – R2D Backup crypto-model
Figure 11 – R2D Emergency crypto-model (a)
Figure 12 – R2D Emergency crypto-model (b)
Figure 13 – R2D Backup sequence diagram
Figure 14 – R2D Emergency sequence diagram
Figure 15 – RDS crypto-model
Figure 16 – RDS sequence diagram



## LIST OF TABLES

Table 1 - Comparison of different types of Blockchain
Table 2 - Comparison of popular blockchain platforms
Table 3 - Notation used

# 1 INTRODUCTION

## 1.1 Scope of the document

The main goal of the present document is to describe the InteropEHRate technical specification of consent management and decentralized authorization mechanisms for health record (HR) exchange focused on all the involved architecture components (e.g., S-EHR App, HCP Web App, S-EHR Cloud, Central Node and Reference Research Center), protocols (e.g., D2D, R2D Access, R2D Backup, R2D Emergency, RDS), and scenarios (e.g., Medical Visit, Emergency and Research). It also provides a detailed description of the security protocols and their functionality.

## 1.2 Intended audience

The current document is mainly intended for developers, architects, manufacturers, security engineers, and all the project participants and partners interested to have an overview of how the InteropEHRate supports consent management and decentralized authorization mechanisms for exchanging health records. Apart from this, the document is intended for researchers as well, as they may be interested in understanding the way that InteropEHRate handles consent management and decentralized authorization and possibly extend and update their specification.

## 1.3 Structure of the document

This deliverable is structured as follows:

- **Section 1** (the current section) introduces the overall concept of the document, defining its scope, intended audience, and relation to the other project tasks and reports.
- **Section 2** describes and reviews the research background of both consent management and decentralized authorization.
- **Section 3** introduces the overall consent management and decentralized authorization mechanisms in terms of InteropEHRate, where it is analysed in detail for all the InteropEHRate protocols. This section includes the security models for all the security protocols to highlight the used crypto-primitives.
- **Section 4** provides a description on the standards considered for the security aspects of InteropEHRate.
- **Section 5** outlines the conclusions of the current document, including the deliverable highlights and the most important aspects of the consent management and decentralized authorization.
- **Appendix A** summarises all the cryptographic notations used for better understanding of the modelling of protocols and the JSON schemas for D2D requests.

## 1.4 Updates with respect to previous version (if any)

Several updates have been made with respect to the previous version. The most important are summarised below:

- Description of all the security models and crypto-primitives regarding consent management and decentralized authorization mechanisms per protocol is included and described in the deliverable.
- Chapter 2 updated with a new section regarding the blockchain background technologies in general and blockchain usage in the health domain.
- The structure of Chapter 3 is completely restructured based on the InteropEHRate protocols for a clearer presentation. In addition, all the security protocols are analyzed in comparison with the previous version of the deliverable.
- Specification has been updated with the inclusion of RDS protocol and a clear distinction between the R2D-based protocols namely R2D Access, R2D Backup and R2D Emergency.
- The section "relations to other deliverables" for similarity with other deliverables has been removed.
- Updated sequence diagrams are provided and described thoroughly in Chapter 3.
- Design of a blockchain-based scheme, as an optional service, for the logging of data sharing transactions is provided.
- A new Chapter added regarding the InteropEHRate alignment with the security-related standards, as suggested in the first review.
- Conclusion section was updated, while no next steps have been included since this is the final version of the deliverable.
- An appendix with all the necessary cryptographic notations of the security models included in the deliverable and the JSON schemas for D2D requests.

# 2 TECHNICAL BACKGROUND

The chapter commenced with a thorough review of the literature to gain a clear understanding of the current state of the art related to the decentralized authorization and consent management. Below it is listed what is authorization and consent management for the sake of completeness.

- **Consent Management (CM)**: is a system, process or set of policies for allowing consumers and patients to determine what health information they are willing to permit their various care providers to access. It enables patients and consumers to affirm their participation in e-health initiatives and to establish consent directives to determine who will have access to their protected health information (PHI), for what purpose and under what circumstances. Consent management supports the dynamic creation, management and enforcement of consumer, organizational and jurisdictional privacy policies **[CM1019]**.
- **Authorization**: is the function of specifying access rights/privileges to resources, which is related to information security and computer security in general and to access control in particular **[FRASER1997]**.

## 2.1 Consent Management

The informed consent of the citizen is essential for data exchange. The EU general data protection regulation 2016/679 (GDPR) forms the legal basis for data processing. Articles 4 (11), 6 (1)(a), 7, 8, and 9(2)(a) and Recitals 32, 33, 38, 42, and 43 of the GPPR deals with the conditions for consent.

The relevant paragraphs and recitals are listed in the following:

> ***Article 4*** *Definitions (11)*
>
> 11. *'consent' of the data subject means any freely given, specific, informed and unambiguous indication of the data subject's wishes by which he or she, by a statement or by a clear affirmative action, signifies agreement to the processing of personal data relating to him or her;*

> ***Article 6*** *Lawfulness of processing (1)(a)*
>
> 1. *Processing shall be lawful only if and to the extent that at least one of the following applies:*
>    a. *the data subject has given consent to the processing of his or her personal data for one or more specific purposes;*

> ***Article 7*** *Conditions for consent*
>
> 1. *Where processing is based on consent, the controller shall be able to demonstrate that the data subject has consented to processing of his or her personal data.*
> 2. *If the data subject's consent is given in the context of a written declaration which also concerns other matters, the request for consent shall be presented in a manner which is clearly distinguishable from the other matters, in an intelligible and easily accessible form, using clear and plain language. Any part of such a declaration which constitutes an infringement of this Regulation shall not be binding.*
> 3. *The data subject shall have the right to withdraw his or her consent at any time. The withdrawal of consent*

*shall not affect the lawfulness of processing based on consent before its withdrawal. Prior to giving consent, the data subject shall be informed thereof. It shall be as easy to withdraw as to give consent.*

4. *When assessing whether consent is freely given, utmost account shall be taken of whether, inter alia, the performance of a contract, including the provision of a service, is conditional on consent to the processing of personal data that is not necessary for the performance of that contract.*

***Article 8*** *Conditions applicable to child's consent in relation to information society services*

1. *Where point (a) of Article 6(1) applies, in relation to the offer of information society services directly to a child, the processing of the personal data of a child shall be lawful where the child is at least 16 years old. Where the child is below the age of 16 years, such processing shall be lawful only if and to the extent that consent is given or authorised by the holder of parental responsibility over the child. Member States may provide by law for a lower age for those purposes provided that such lower age is not below 13 years.*
2. *The controller shall make reasonable efforts to verify in such cases that consent is given or authorised by the holder of parental responsibility over the child, taking into consideration available technology.*
3. *Paragraph 1 shall not affect the general contract law of Member States such as the rules on the validity, formation or effect of a contract in relation to a child.*

***Article 9*** *Processing of special categories of personal data (1), (2)(a)*

1. *Processing of personal data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, or trade union membership, and the processing of genetic data, biometric data for the purpose of uniquely identifying a natural person, data concerning health or data concerning a natural person's sex life or sexual orientation shall be prohibited.*
2. *Paragraph 1 shall not apply if one of the following applies:*
    a. *the data subject has given explicit consent to the processing of those personal data for one or more specified purposes, except where Union or Member State law provide that the prohibition referred to in paragraph 1 may not be lifted by the data subject*

***Recital 32*** *Conditions for consent*

*Consent should be given by a clear affirmative act establishing a freely given, specific, informed and unambiguous indication of the data subject's agreement to the processing of personal data relating to him or her, such as by a written statement, including by electronic means, or an oral statement. This could include ticking a box when visiting an internet website, choosing technical settings for information society services or another statement or conduct which clearly indicates in this context the data subject's acceptance of the proposed processing of his or her personal data. Silence, pre-ticked boxes or inactivity should not therefore constitute consent. Consent should cover all processing activities carried out for the same purpose or purposes. When the processing has multiple purposes, consent should be given for all of them. If the data subject's consent is to be given following a request by electronic means, the request must be clear, concise and not unnecessarily disruptive to the use of the service for which it is provided.*

***Recital 33*** *Consent to certain areas of scientific research*

*It is often not possible to fully identify the purpose of personal data processing for scientific research purposes at the time of data collection. Therefore, data subjects should be allowed to give their consent to certain areas of scientific research when in keeping with recognised ethical standards for scientific research. Data subjects should have the opportunity to give their consent only to certain areas of research or parts of research projects to the extent allowed by the intended purpose.*

***Recital 38*** *Special protection of children's personal data*

*Children merit specific protection with regard to their personal data, as they may be less aware of the risks, consequences and safeguards concerned and their rights in relation to the processing of personal data. Such specific protection should, in particular, apply to the use of personal data of children for the purposes of marketing or creating personality or user profiles and the collection of personal data with regard to children when using services offered directly to a child. The consent of the holder of parental responsibility should not be necessary in the context of preventive or counselling services offered directly to a child.*

***Recital 42*** *Burden of proof and requirements for consent*

*Where processing is based on the data subject's consent, the controller should be able to demonstrate that the data subject has given consent to the processing operation. In particular in the context of a written declaration on another matter, safeguards should ensure that the data subject is aware of the fact that and the extent to which consent is given. In accordance with Council Directive 93/13/EEC[1] a declaration of consent pre-formulated by the controller should be provided in an intelligible and easily accessible form, using clear and plain language and it should not contain unfair terms. For consent to be informed, the data subject should be aware at least of the identity of the controller and the purposes of the processing for which the personal data are intended. Consent should not be regarded as freely given if the data subject has no genuine or free choice or is unable to refuse or withdraw consent without detriment.*

***Recital 43*** *Freely given consent*

*In order to ensure that consent is freely given, consent should not provide a valid legal ground for the processing of personal data in a specific case where there is a clear imbalance between the data subject and the controller, in particular where the controller is a public authority and it is therefore unlikely that consent was freely given in all the circumstances of that specific situation. Consent is presumed not to be freely given if it does not allow separate consent to be given to different personal data processing operations despite it being appropriate in the individual case, or if the performance of a contract, including the provision of a service, is dependent on the consent despite such consent not being necessary for such performance.*

In addition to the GDPR which applies to all European Union countries, there may also be country-specific regulations.

Legally, the consent acquired from citizens must contain the following building blocks according to Art. 7 of the GDPR:

- It must be clearly written in simple and plain language, understandable to the citizen and must be provided on a separate form or if not, the consent must be clearly distinguishable from other matters on the form which the citizen will sign
- Demographic data of the citizen
- The identity of the controller (Recital 42)
- A statement about the citizen's right to withdraw consent and that consenting to the processing is not a condition for the performance of any contract, information about the citizen's privacy protection rights and/or data exchange processing (Recital 42)
- Separate consent must be possible to be given to different personal data processing operations (Recital 43). Consent should cover all processing activities carried out for the same purpose or purposes (Recital 32). When the processing has multiple purposes, consent should be given for all of them (Recital 32).
- General information about the data exchange and/or processing
    - type and purpose of electronic data exchange and/or processing (if the data exchange or processing has multiple purposes, consent should be given for all of them)
    - scope of data exchange (information objects which will be exchanged)
    - list of access permissions (for example, hospitals, medical care centres)
- Citizen's declaration
    - of voluntary and informed consent to the specified data processing activities for the specified consent and time period
    - that he or she received information about withdrawal of consent, privacy protection rights and/or data exchange processing

Please also note that Articles 13 and 14 require that specific information be given to the data subject whose personal data is being processed.

### 2.1.1 Existing Standards for structural representation of consents

The IHE Advanced Patient Privacy Consents (APPC) Profile **[APPC]** defines a structural and semantic representation of a privacy consent policy to enable consent(s) to be captured, managed and exchanged between systems. The aim of the APPC Profile is to enforce interoperability between access control systems and to support system-wide authorization mechanisms.

The profile defines two actors:

- Content Creator: system which creates a structured machine-readable consent document.
- Content Consumer: system which consumes a structured consent document. This system shall be able to process and interpret the structured policies contained in the APPC consent document.

The following sequence diagram illustrates this process (Figure 1). The presentation is generic and does not specify how the consent document is transmitted. The Content Creator (as part of healthcare related application) captures all the information needed for consent creation. In cross-facility supply scenarios, the unique identification of service providers, such as through the provider information directory service, is essential. The unique ID can be used to check whether there is a right of access for the organization/person with this ID. The resulting consent document is transferred to systems that implement access management. The content consumer can be an extension of these systems.

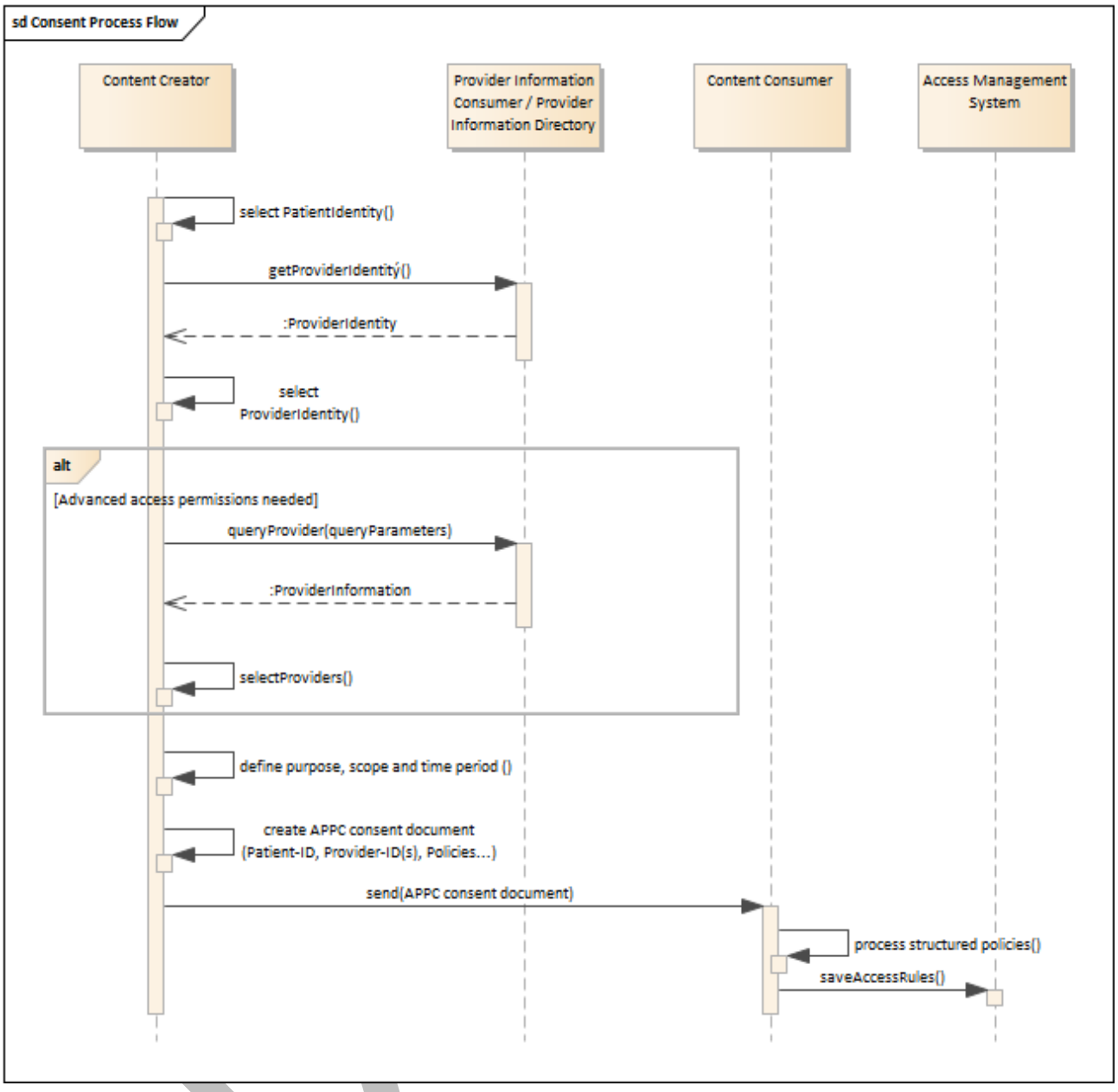



*Figure 1 - Consent Process Flow*

The structure of the APPC consent document is based on XACML **[XACML]**. Details to the structure of the policies will be described in the following sections.

Another profile is the IHE Basic Patient Privacy Consents (BPPC) **[BPPC]**. However, it is not described in more detail due to the lack of expressiveness of access rules.

## 2.2 Decentralized Authorization

In decentralized authorization, the decision to grant or deny access is based on two distinct processes, authentication and authorization. Authentication involves the verification of credentials, whereas

authorization is the process of granting or denying access to system resources based on credentials. Deliverable **[D3.4]** summarizes a detailed state-of-the-art of the identification and authentication mechanisms, while this deliverable handles access control, which is one of the main methodologies used to perform the verification of the authorization of an end user requesting access to specific restricted resources.

In the literature, many commonly used authorization/access control models are defined. The best-known are Mandatory Access Control (MAC), Discretionary Access Control (DAC) and Role-Based Access Control (RBAC). All these models are known as identity-based access control models where users (subjects) and resources (objects) are identified by unique names. Static access control models usually provide a list of permissions that each subject has on certain objects. The literature on combining context and security mainly concentrates on context-based RBAC. In addition, in the literature, a fourth type has been identified, the Attribute Based Access Control (ABAC). Such a scheme is by nature dynamic. The main difference of ABAC with the previous schemes is the fact that the concept of provided policies can express a complex boolean rule set that can evaluate many different attributes. In ABAC, there are not static lists of permissions that associate subjects with objects, but instead there are "snapshots" of such associations that can be generated and dynamically changed based on the current context. Any ABAC system should implement the conceptual flow that is depicted in Figure 2 below.

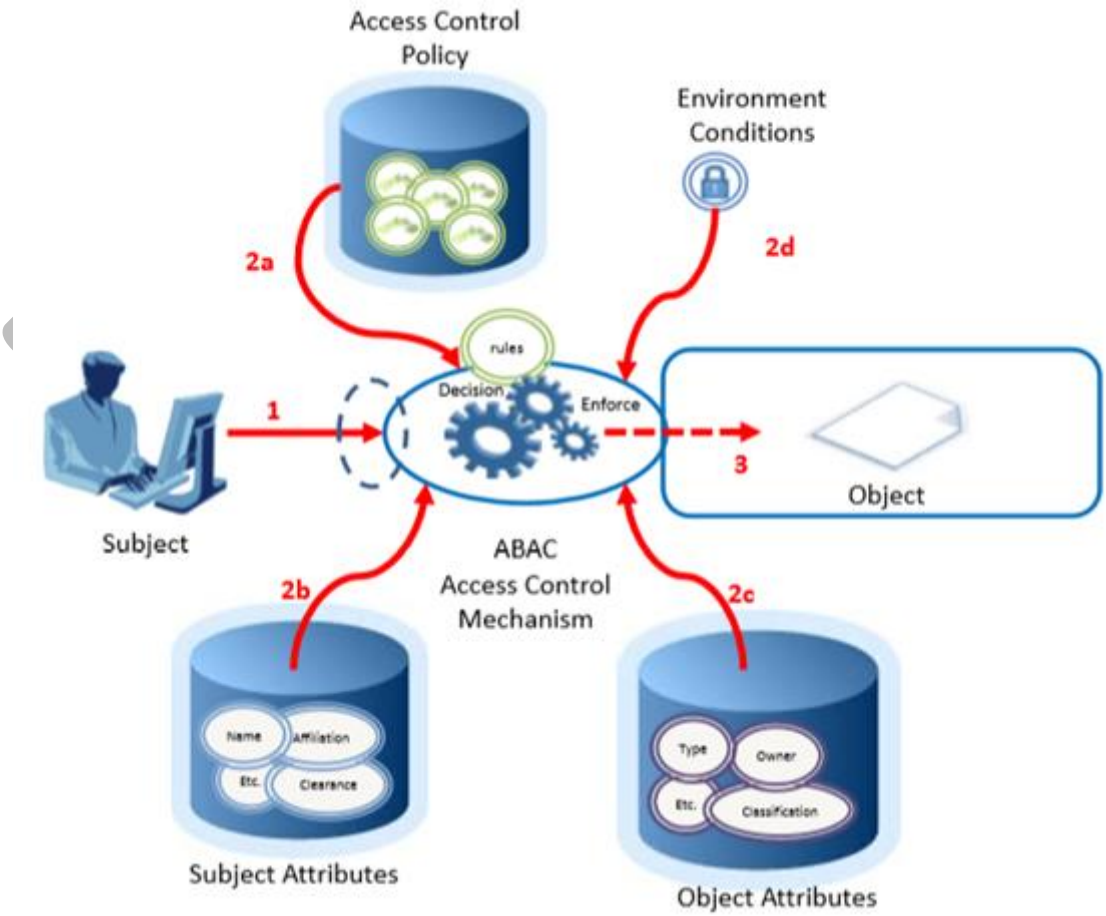



*Figure 2 - ABAC Indicative Information Flow*

The access request (step 1) is handled by the ABAC Access Control Mechanism which consults a policy repository (step 2a) in order to obtain the set of attributes that have to be examined in order to reach a decision of “allow” or “deny”. The attribute examination phase checks subject attributes (step 2b), object attributes (step 2c) and environmental attributes (step 2d) in order to perform the actual assessment (step 3). In general, ABAC as the most prominent access control mechanism, avoids the need for capabilities to be directly assigned to subject requesters or to their roles or groups before the request is made. Instead, when a subject requests access, the ABAC-compliant engine can make an access control decision based on the assigned attributes of the requester, the assigned attributes of the object, environment conditions, and a set of policies that are specified in terms of those attributes and conditions.

### 2.2.1 Existing ABAC Standards

As already discussed, there are many reference implementations of the ABAC model. One example of an access control framework that is consistent with ABAC is the eXtensible Access Control Markup Language (XACML) **[XACML]**. Another example is the Next Generation Access Control (NGAC) standard **[NGAC]**. These two are considered to be the most notable ones **[FERRAIOLO2016]**.

#### *2.2.1.1 XACML in a Nutshell*

XACML is an OASIS **[XACML]** standard that describes both a policy language and an access control decision request/response language. Both languages use XSD notations; hence policy definition and request/response elements are serialized as XML elements. The policy language details general access control requirements, and has standard extension points for defining new functions, data types, combining logic, etc. The request/response language lets you form a query to ask whether or not a given action should be allowed, and interpret the result. The response always includes an answer about whether the request should be allowed using one of four values: Permit, Deny, Indeterminate (an error occurred or some required value was missing, so a decision cannot be made) or Not Applicable (no policy available to this service addresses this request). In the context of InteropEHRate an XACML-based ABAC mechanism is adopted for healthcare professionals authorization to download citizens’ medical data in emergency situations.

The specification defines five main components (Figure 3) that handle access decisions; namely Policy Enforcement Point (PEP), Policy Administration Point (PAP), Policy Decision Point (PDP), Policy Information Point (PIP), and a Context Handler.

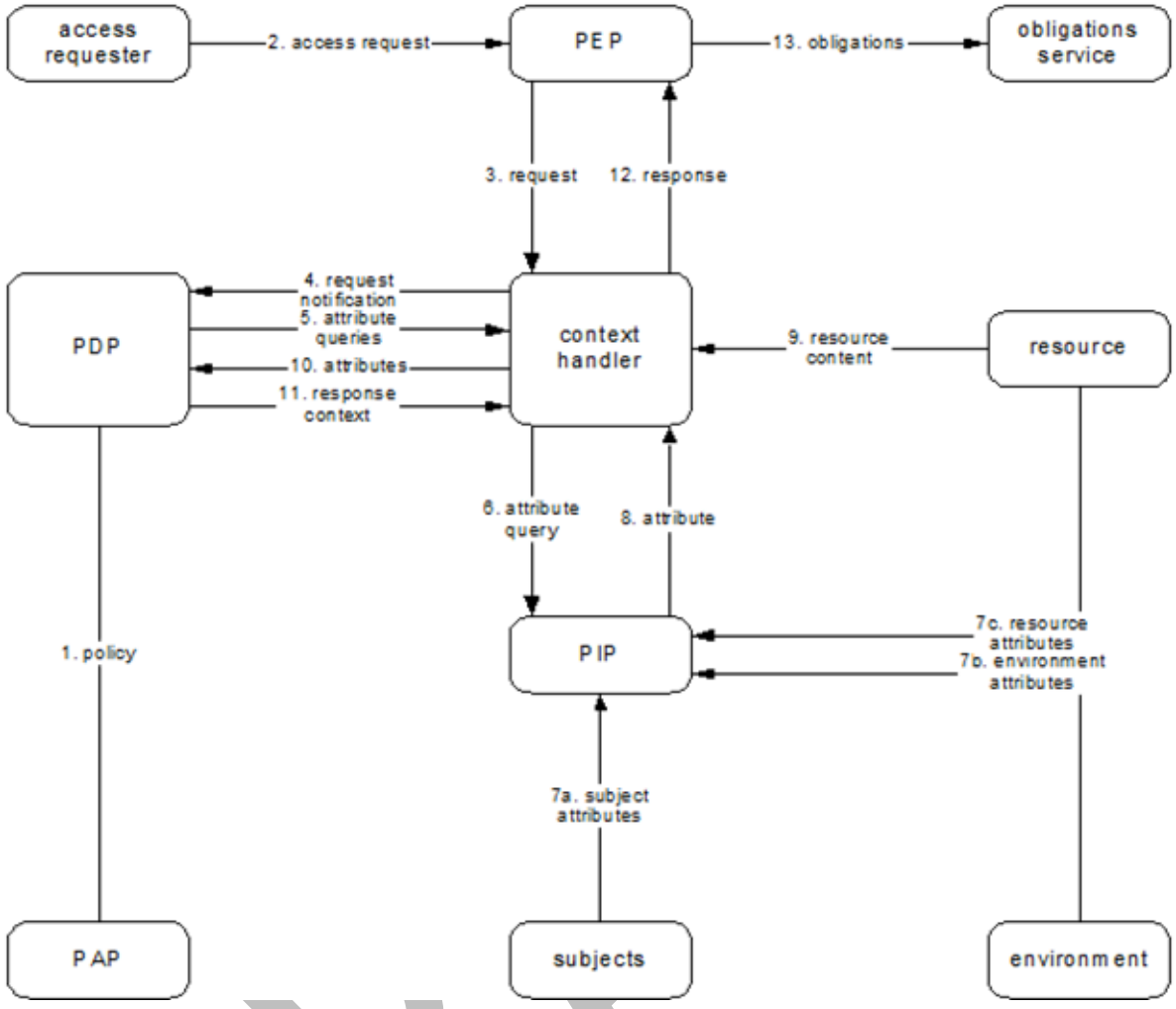



*Figure 3 - XACML data-flow diagram*

The functional purpose of the main components is:

- The Policy Administration Point (PAP) provides an interface or API to manage the policies that are stored in the repository and provides the policies to the Policy Decision Point (PDP).
- The Policy Enforcement Point (PEP) is the interface to the external world. It receives the application specific access requests and translates them to XACML access control requests, then it denies or allows access based on the result provided by the PDP.
- The Policy Decision Point (PDP) is the main decision point for the access requests. It collects all the necessary information from other actors and yields a decision.
- The Policy Information Point (PIP) is the point where the necessary attributes for the policy evaluation are retrieved from several external or internal actors. The attributes can be retrieved from the resource to be accessed, environment (e.g. time), subjects, and so forth.
- The Context Handler entity converts decision requests in the native request format to the XACML canonical form, coordinates with Policy Information Points to add attribute values to the request context, and converts authorization decisions in the XACML canonical form to the native response format.

#### 2.2.1.2 NGAC in a Nutshell

NGAC **[NGAC]** is a fundamental reworking of traditional access control into a form that suits the needs of the modern distributed interconnected enterprise. NGAC diverges from traditional approaches to access control in defining a generic architecture that is separate from any particular policy or type of policy. NGAC is not an extension of, or adaptation of, any existing access control mechanism, but instead is a redefinition of access control in terms of a fundamental and reusable set of data abstractions and functions. NGAC provides a unifying framework capable without extension of supporting not only current access control approaches, but also novel types of policies that have been conceived but never implemented due to the lack of a suitable enforcement mechanism.

This standard contains an abstract functional description of an architecture. The description is abstract because it excludes all irrelevant details, and is functional because it partitions the entities comprising the architecture purely on the basis of their function and excludes all other constraints. NGAC does not express policies through rules, but instead through configurations of relations of four types: assignments (define membership in containers), associations (to derive privileges), prohibitions (to derive privilege exceptions), and obligations (to dynamically alter access state). The specification defines six main components (Figure 4) that handle access decisions; namely Policy Enforcement Point (PEP), Resource Access Points (RAP), Policy Decision Point (PDP), Policy Access Point (PAP), Policy Information Point (PIP), and optional Event Processing Point (EPP).

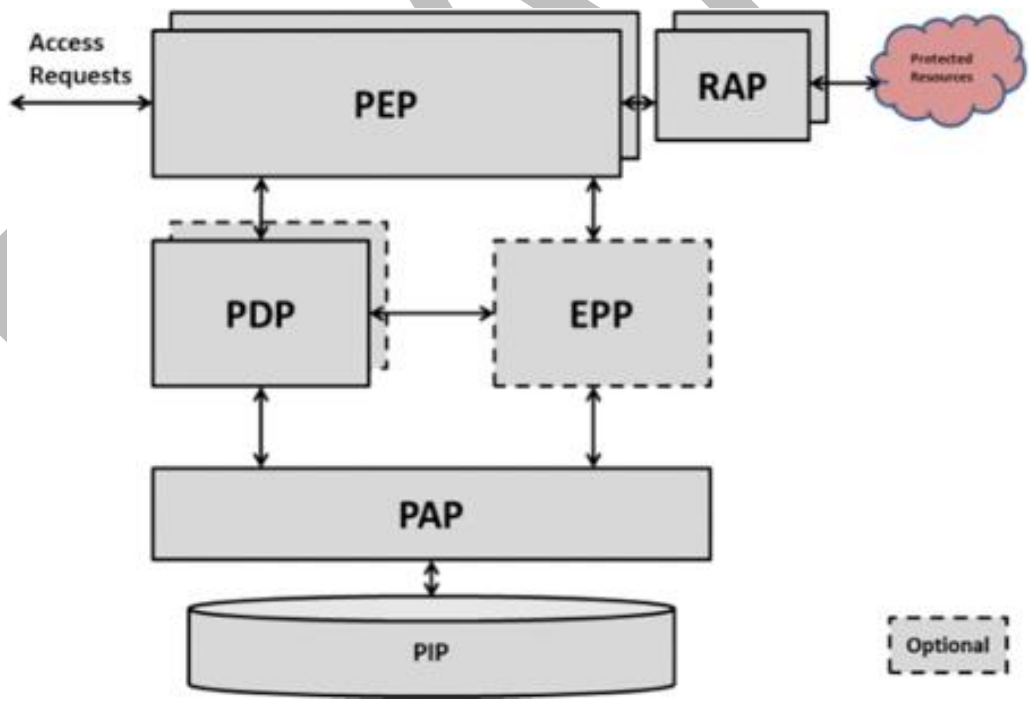



*Figure 4 - NGAC data-flow diagram*

## 2.3 Blockchain and Distributed Ledger Technologies

A distributed ledger technology (DLT) is a type of data structure which is replicated, shared, and synchronized across multiple devices that may operate geographically distant from each other. Data synchronization in a distributed ledger is achieved by using a consensus mechanism among the parties that

hold a copy of the ledger. More specifically, consensus refers to a protocol that ensures that parties agree to a specific state of the system as the valid one. Different types of distributed ledgers have been defined over the last years with blockchain being the most prominent one. The following sections provide a literature review on the blockchain usage in the healthcare domain, the necessary background on the blockchain technology and the InteropEHRate vision of the blockchain adoption as an optional service.

### 2.3.1 Blockchain Usage in Healthcare

Blockchain technology has the ability to transform healthcare by placing the patient at the center of the health system and at the same time to increase the security, privacy, and interoperability of health data. Blockchain is considered a useful technology for managing sensitive data, especially for the sectors of healthcare, medical research and insurance **[Tandon2020]**. In parallel, the increasing digitization in the healthcare sector has led to the acknowledgment of concerns related to secure data storage, data ownership and sharing of medical data **[Meinert2019]**. Even though blockchain usage in the healthcare domain has added significant value through improved efficiency, access control, technological advancement, privacy protection, and security of data management processes, the adoption is relatively slow. Until now in the literature, several research works suggested blockchain-based solutions for the critical challenges faced by the healthcare domain. Blockchain, by its nature, can protect healthcare data from potential data loss, corruption or attack due to the replicated information among multiple nodes.

The ability of blockchain to preserve and record data (data management) and extract knowledge from the recorded data (knowledge extraction) is the main reason behind the use of blockchain in healthcare **[Kuo2019]**. In addition, the immutability property - non changeable stored data in a block due to unintentional or malicious reasons - of blockchain aligns very well with the requirements for medical data storage. Moreover, blockchain enables data records to be unified, updated, securely exchanged, and accessed in a timely manner by the appropriate authorities. This is a major advantage of blockchain technology within the healthcare domain, since current practices require data to be stored with third parties **[Hölbl2018]**. In addition, blockchain has the potential to address critical concerns, such as automated claim validation **[Angraal2017]** and public health management **[Mettler2016]**. Last but not least, blockchain can bring transparency to data management **[Ito2018]**. Despite the advantages of the effect of the blockchain on societal and business transformation, there seems to be a debate on its prevalent advantages and benefits in comparison to previously established expectations **[Tandon2020]**. According to **[Tandon2020]** a recent report suggests that in the future organizations will likely adopt a cautiously pragmatic blockchain-based approach because of a prevalent belief that the benefits may be over-hyped. This may be additionally compounded by a general uncertainty about usage of the blockchain with respect to legal compliance and government regulations **[Alla2018]**. Healthcare domains that could profit from the blockchain adoption according to **[Biplov2020]** are listed below:

- **Personal health record storage**. EMRs could be realised in blockchain where personal health records can be stored. Also, using blockchain as a repository for EMRs could give rise to universal and interoperable data format **[Swan2015]**.
- **Blockchain Health Research Commons**. Blockchain technology could provide a model for establishing a cost-effective public-health data commons. Individuals would like to contribute personal health data (e.g. FitBit, MapMyRun etc.) to research commons, but until now a venue does not exist.

- **Blockchain health notary**. Providing proof-of-existence documents which are usually carried out by notaries can also be incorporated in the blockchain. Health related documents could be encoded in blockchain which could be verified and confirmed in seconds by the required party thus removing the need for paper as a proof.
- **Doctor Vendor Request for Proposal Services and Assurance contracts**. Such a service could help facilitate the price transparency between doctor and patient treatment process.

**InteropEHRate vision** is to use a permissioned private blockchain for secure, privacy-preserving and auditable data transaction logging mechanism as an optional and complementary service, where each research center may have its own private blockchain. In addition, there is a need to keep logs in an auditable manner of all the data management transactions. Such a logging mechanism, with the information of who shared the data (in an anonymous way), who has access to the data, and when the transaction is performed, is of paramount importance especially in the healthcare domain to keep track of the provenance history. This mechanism works in tandem and complements the existing state of the art for data management and knowledge extraction among actors. On top of that, blockchain offers a plethora of features, such as traceability, transparency, automation, decentralization, and security. Despite these promising features, the technical scalability of the network is still a key barrier which can put a strain on the adoption process, especially for healthcare environments. Throughput, storage, and networking are three aspects of scalability that should be considered to improve InteropEHRate network scalability. InteropEHRate blockchain is based on Hyperledger Fabric that is a stable and well tested system for data transactions and has a cost-effective network for the operation of its applications.

### 2.3.2 Blockchain Basic Elements, Types and Consensus Algorithms

The name blockchain stems from its technical structure — a chain of blocks **[Wüst2018]**. A blockchain is an append-only ledger database organized as a chain of blocks that relies on a peer-to-peer network to perform its management, updates, and operations. A block is a collection of valid transaction proposals that are received within a period of time. The first block is called a “genesis block” and each individual block consists of a block header and the block body. The header includes the block version, a parent block hash, a Merkle tree root hash, a timestamp, nBits and nonce, while the body is filled with the transaction history. In order to enable an entity to write information on the Blockchain, a party is needed who summarizes information and records it into the block. The chaining of blocks is achieved through the usage of hash functions. Also, a communication network enables peers to exchange information, transaction, ownership, and maintain elements of cryptography for immutability and security via the use of cryptographic primitives and consensus algorithms (e.g., Proof of work, Proof of stake).

The corresponding read permission is controlled by the access layer that the Blockchain will allow a different access control over the read rights. Blockchains can be classified based on: (a) accessibility, (b) consensus mechanisms, and (c) its crypto currency **[Bocek2018].** In the literature three different categories of blockchain in the accessibility exist (access layer): a) **Permissionless blockchain**. This type of blockchain is public and decentralized and anyone can query. It is also open to anyone who wants to be part of the blockchain processing services by running the consensus protocol with the proper hardware. This openness implies that the stored content is readable by any peer. The most known permissionless blockchains are the Bitcon and the Ethereum. b) **Public permissioned blockchain**. This type of blockchain is also known as consortium blockchain and is a hybrid implementation between a permissionless and a private

permissioned blockchain. In this type anyone can be allowed to read the blockchain state (public verifiability). Two or more nodes are permitted to take part in the mining process -solving complex mathematical problems-, in the sense that only authorized users can be part of the network. Therefore it has the advantages of decentralization and the improved security and privacy inherent in the private blockchain. c) **Private permissioned blockchain**. In this type of blockchain each network user (e.g. in the context of InteropEHRate researchers that can have access to the history of the logs) must be enrolled with a central authority before joining it. Related applications include database management, auditing, etc. The most known permissioned blockchains are Hyperledger Fabric and Corda. In this type a restricted set of readers is allowed to read the blockchain state. Only one node is permitted to take part in the mining process.

Table 1 summarizes the pros and cons of all the three types of Blockchain. It can be seen that permissionless blockchain provides the best accessibility and full immutability. However it seems that it may not be the best candidate for healthcare data storage with a limited number of parties and partners of an organisation. At the same time the ledger information may not be of benefit to become fully public. As for a private permissioned blockchain, it appears to be a perfect match for the InteropEHRate, as it requires access control over authenticated parties, it is highly efficient and with fast transactions. Accordingly, the public permissioned blockchain may provide better flexibility and adjustability on security, privacy and regulation. It seems to offer a middle path between public and private blockchains.

| | Permissionless blockchain | Private Permissioned blockchain | Public Persmissioned blockchain |
|---|---|---|---|
| Access | Anyone | Authenticated users | Mixed users |
| Authority | Decentralized | Partial decentralized | Mixed |
| Transaction Speed | Slow | Fast | Adjustable |
| Efficiency | Low | High | Flexible |
| Data handling | Read and write access for anyone | Read and write for authenticated users | Mixed |
| Immutability | Full | Partial | Flexible |

*Table 1 - Comparison of different types of blockchain*

**Consensus mechanisms** in distributed systems have been a well-studied research problem the last three decades. Consensus algorithms can be directly applied to block mining, transaction verification and any on-chain actions requiring the "voting" of all/partial nodes within the network. Some of the most known consensus algorithms in the literature are listed below **[Oyinloye2012]**.

- **Proof of Work (PoW)** is one of the first and most widely adopted consensus protocols used in blockchain applications and requires a prover and a verifier to compute a hash puzzle and check if the puzzle is correct, respectively.
- **Proof of Stake (PoS)** avoids the energy consumption, and penalty shortcomings similarly to PoW. A participating entity must have some stake (e.g. cryptocurrency) in the system in order to mine or validate block transactions. Its core idea is to choose a block creator via various combinations of random selection based on the amount of owning currencies.
- **Practical Byzantine Fault Tolerance (PBFT)** is based on a voting process in order to add the block. Consensus is based on replication between known parties that can tolerate a failure of up to one-third of the parties

### 2.3.3 Smart Contracts

The concept of legal contracts in the form of paper that are enforced and verified by intermediaries has been around for a long time. A smart contract is a computer program **[Szabo1997]** which is intended to automatically execute, control or document legally relevant events and actions according to the terms of a contract or an agreement. These programs can sit inside the block of a blockchain and can be digitally invoked avoiding intermediaries in a decentralized manner. When the smart contract detects the fulfilment of a pre-programmed condition, it executes the corresponding action. Smart contracts are independently and autonomously executed by the blockchain and as an actual part of the blockchain are immutable and transparently stored on the ledger. In this way the execution of a contractually specified action cannot be prevented or manipulated. Smart contracts can interact with each other, and are triggered by events in the real world. As an example, smart contracts are being used nowadays for: a) managing authorship and implementing pay-per-use systems in digital works; b) automatic payments for goods and services; c) life insurance, vehicles which payment depends on the use of the active contract; d) IoT devices and machines exchanging data for money; and e) registering the user in renting processes.

The first generation of smart contracts was a traditional contract but adding a common logic for every involved party and a common verification mechanism, protected with cryptographic protocols. The most known cryptocurrencies have smart contracts for the mining behaviour, transaction fees or even withdrawal limits **[Nakamoto2008]**. However, smart contracts cover more use-cases than a coin exchange, ranging from financial contracts to gambling. More recently, smart contracts have been extended to other payment-related domains. Offering automation, transparency, traceability and tamper proof transactions, blockchain-based smart contracts have been becoming popular in the deployment of the sectors like government, healthcare and the real estate industry **[Li2019]**. Typical examples are supporting quick response operations in supply chain **[Li2019]**, compiling the control flow and business logic, facilitating real-time order settlement of manufacturing **[Sheel2019]**, and providing hyperconnected logistics **[Betti2019]**. In addition, smart contracts can be beneficial to the healthcare sector by monitoring medicine selling, medical payment transactions, tracing the status of drugs **[Clark2018]**, while smart contracts can be used to define authorization rules that allow the patients to be in control of how their health records are used **[Swan2015]**.

Last but not least, smart contracts are the perfect tool to do the security enforcement process, as it is a transparent, auditable code, which can be reviewed by anyone but changed by no-one, and automatically controls the logic of the enforcement process without any human intervention **[Saad2020]**. Moreover,

smart contracts and ABAC authorization based on data access control with ABE are closely related concepts; this is mainly based on the fact that both provide data access based on attributes and policies. ABE is not an authorization mechanism but a cryptographic primitive that allows multiple users to encrypt and decrypt files based on their attributes and encryption policies. On top of that, an important part of the ABE is the verifiability of attributes.

### 2.3.4 Popular Blockchain Platforms

Inspired by the renowned 1st generation public blockchain platforms, e.g. Bitcoin and Ethereum, there have been many 2nd gen or 3rd generation blockchain platforms developed for various real-world applications. According to **[Forbes2019]** Hyperledger, Ethereum, Quorum and R3 Corda are considered as the most popular blockchain implementations:

- **Hyperledger Fabric** is an enterprise-grade permissioned distributed ledger framework backed by the Linux Foundation and IBM among others. It is a decentralised operating system for permissioned blockchains that can execute distributed applications (Dapps) written in general-purpose programming languages such as Go, Java or Node.js **[Androulaki2018]**. There are three types of nodes in Hyperledger Fabric: a) client, b) peer and c) ordering service nodes. In the context of InteropEHRate this permissioned framework is adopted.
- **Ethereum**, is an open source, public, blockchain-based, distributed platform for developing decentralised applications **[Ethereum2020]**. Originally, Ethereum was a public permissionless blockchain platform implementing a Proof-of-Work (PoW) based consensus protocol called Ethash. Ethereum is also used as a private platform as a configurable feature. Ethereum is structured as a peer-to-peer network where peers which ensure network transactions verification are called nodes.
- **Quorum** has been developed by J.P. Morgan for financial use-cases, however, can be used for any other type of industry use caseS. It is a permissioned blockchain based on the Ethereum blockchain and more specifically, is a fork of the go-ethereum implementation **[Quorum2021]**. Quorum node is an Ethereum Geth node which has been modified to additionally handle private transactions.
- **R3 Corda** was one of the first proposals for enterprise blockchain solutions targeting financial services. Corda is an open source permissioned platform developed by R3 **[Corda2021]**. It follows the "Know Your Customer" principle, each node has to prove its identity to be authorised to join the network.

The previously introduced blockchain platforms have respective pros and cons. In the InteropEHRate project, in order to provide a better self-inclusive and self-maintain environment for data storage, it is better to focus on permissioned platforms, e.g., Quorum, Hyperledger Fabric or R3 Corda, aiming to open source platforms while promoting support for smart contracts. Table 2 presents a brief comparison for the current popular Blockchain platforms related to efficiency, security and other user-driven requirements. Among these platforms, it is observed that Hyperledger Fabric is the best candidate, since it is a stable, cross-industry focused, with user authentication and data confidentiality and with instantaneous transactions per second, which is a highly desirable feature for the healthcare domain.

| Ledger type | Hyperledger Fabric | Ethereum | Quorum | R3 Corda |
|---|---|---|---|---|
| **Industry focus** | Cross-industry | Cross-industry | Cross-industry | Financial service |
| **Ledger type** | Permissioned | Permissionless | Permissioned | Permissioned |
| **Consensus algorithm** | Raft, BFT, and Pluggable support | PoW, PoS | Majority voting Raft, BFT, PoAuthority | Raft, BFT |
| **Smart contract support** | ✓ | ✓ | ✓ | ✓ |
| **Programmable And open source** | ✓ | ✓ | ✓ | Smart contract support under development |
| **Transaction confirmation time** | Instantaneous | around 12 seconds | Instantaneous | Instantaneous |
| **Identity** | Identified | Pseudo-anonymous | Identified | Identified |
| **Data confidentiality** | ✓ | ✕ | Using private channel | Within transaction group |
| **User authentication support** | ✓ Enrolment certificate | ✓ Via digital signature | ✓ Address from public key | ✓ Enrolment certificate |

*Table 2 - Comparison of popular blockchain platforms*

## 2.4 Relation with other research projects

This section presents other complex representative eHealth projects which can be considered as having similarities with InteropEHRate, such as:

- cybersecurity in healthcare;
- eHealth / mHealth;
- HL7 compliance;
- standardization specific to the eHealth field.

The InteropEHRate project benefits from the relevant experience, specific results and know-how acquired by these projects. All projects below target the healthcare sector and address the same categories of key stakeholders, such as:

- industry (suppliers of digital wearables, suppliers of communication devices, suppliers of IT / eHealth solutions and services, suppliers of cyber security services);
- healthcare organizations;
- research organizations;
- policymakers;

- governmental organizations and agencies;
- patients organizations / associations;
- international networks (eHealth, cybersecurity, interoperability, standardization).

Moreover, all projects share the use of innovative technologies, technological platforms and software tools in the healthcare field, both in implementing the integrated IT solutions as well as ensuring interoperability with external systems developed on the same principles (compliant with HL7 / HL7 FHIR).

Concerning the benefits generated by the synergy among InteropEHRate and the projects mentioned below, InteropEHRate caught up from these projects those results and the know-how necessary for the development of the platform as well as the specific business information regarding the business models and the marketing approach when exploiting the results of the project.

InteropEHRate, also benefited from the following specific results of the other projects, namely: state-of-the-art aspects, collaboration for the development of an effective interoperability framework in the healthcare sector, user needs and specifications identified by each project, methodological approaches specific to eHealth, how to develop a synergy between various projects funded by the H2020 programme and the AAL programme.

We believe that the above mentioned aspects enhance the sustainability impact of the InteropEHRate project and bring added value to Exploitation, Dissemination and Communication specific activities.

### 2.4.1 LETITFLOW - Active Distributed Workflow System For Elderly (AAL Programme)

LetItFlow provides an innovative solution to support elderly hospital staff to accomplish their daily tasks optimally and to adopt and adapt to new procedures and methods via real-time context aware tools. LetItFlow combines two challenges: change management and workflow technologies dedicated to elderly employees. The solution enables assistance by means of tracking employees' activity, portable communication tools, alarming, alerting and notifications services, adapted interaction interfaces or workflow management. It is based on fixed and mobile platforms that interact with the employees to guide them in their work activities. To address the specific requirements from the nurses in the ward, a dedicated interface for the mobile platform was designed based on LetItFlow solution together with new features such as a bed/room map, patient information, complementary tests, shift change reports and patient periodic reports.

LetItFlow proposes methods and tools to facilitate the adaptation of elderly nurses to changeable work environment by supporting them with real-time context aware tool for guiding them in daily tasks. The final objective is to retain the older adult nurses, to avoid their demotivation and ease their daily works, to facilitate knowledge transfer, to increase their efficiency and safety issues. The LetItFlow tool alleviates the task of nurses by a better distribution of the workload. The tool also supports knowledge transfer in real life scenarios, by putting together young and elderly employees in several works.

### 2.4.2 SPHINX – A Universal Cyber Security Toolkit for Health-Care Industry (H2020)

The H2020 SPHINX project aims to introduce a health tailored Universal Cyber Security Toolkit, thus enhancing the cyber protection of the Health and care IT Ecosystem and ensuring patient data privacy and integrity. It will also provide an automated zero-touch device and service verification toolkit that will be easily adapted or embedded on existing, medical, clinical or health available infrastructures. Hospitals and care centres store and exchange large amounts of sensitive patients' data, so they are prime targets for cyber criminals. Since 2016 the published number of health records to have been stolen has been over 2.5 Million, this could be much higher. This varies from inside job attacks, poor security and hacking. At the same time medical devices and wearable devices collecting personal data, become more sophisticated and connected and the use of smartphones makes the whole health system more vulnerable. The health system has to face advanced persistent threats such as Ransomware, Human Threats, DdoS, Lost Info, active attacks and much more.

The SPHINX Toolkit will be validated through pan-European demonstrations in three different scenarios at different countries (Romania, Portugal and Greece). Hospitals, care centres and device manufacturers participating in the project's pilots will deploy and evaluate the solution at business as usual and emergency situations across various use case scenarios.

### 2.4.3 Blockchain usage in healthcare projects

Some known projects in the healthcare domain that utilize blockchain technology follow. GemHealth is a network developed by Gem that enables application development and shared infrastructure for healthcare **[Prisco2016]**. More specifically, this blockchain network includes identity schemes, data storage, and smart contracts applications that execute against shared data infrastructure. GemHealth provides an ecosystem for exchanging enterprise data in a peer-to-peer fashion, both within and across organisations, while creating unique global identifiers for the data assets such that they can easily be tracked between systems **[Allison2017]**. Another open source blockchain-based project is MedRec, that primarily targets patient agency, providing a transparent and accessible view of medical history **[Azaria2016]**. MedRec facilitates the management of permissions, authorization and data sharing between healthcare entities. It is based on the use of Ethereum smart contracts that link patients and providers to the addresses of existing data records. Patient's health data is not stored directly on the Blockchain. Only encoded metadata is stored to allow secure access to data that is securely stored off-chain. The metadata contains information about ownership, permission and the integrity of the data being requested, thus, also enabling efficient data querying. In addition, Philips research worked on "verifiable data exchange", to enable researchers in a network of hospitals and universities to request data that match their need for research purposes **[Dickson2018]**. The project aims to record all data exchanges and the identity of people performing those exchanges in the institutions doing those exchanges. By the term "verifiable data exchange" is implied that the control of the actual audit trail of the request and the fulfilment of the request for data recorded is on the involved institutions. Transparency in data storage and data exchange between involved parties is necessary to create a system of shared risk and responsibility, according to philips researchers. More recently, Guardtime and Estonian biobank deployed a blockchain based data access and governance service for medical data **[Guardtime2019]**. Their blockchain-based MyPCR platform can be accessed through smart phones by 30 million NHS patients **[Guardtime2019b]** to validate patients' identities for the citizens of Estonia. Medicalchain project, blockchain-based platform facilitates the sharing of patients' medical records

across international healthcare institutions, and the Healthcoin initiative, which aims at constructing a global EMR system. Last but not least, other healthcare projects based on blockchain include Factom, HealthCombix, Patientory, SimplyVital, IBM's Watson, BurstIQ, Bowhead, QBRICS and Nuco [Engelhardt2017].

# 3 INTEROPEHRATE CONSENT MANAGEMENT AND DECENTRALIZED AUTHORIZATION

The purpose of this section is to present how the InteropEHRate project will handle consents and authorization for all the protocols and use cases. The following subsections include the crypto models for consent management and decentralized authorization for all communication channels and involved applications. Also, an overview of how the different actors and organizations involved in the InteropEHRate architecture in **[D2.6]** interact with each other is depicted in Figure 5 for better understanding of the different protocols and security needs. More specifically, the InteropEHRate architecture involves the following communication protocols: the device-to-device (D2D), the remote-to-device Access (R2D Access), the remote-to-research Access (R2R-Access) which is similar to R2D Access as an optional extension of the RDS protocol, the remote-to-device Backup (R2D Backup), the remote-to-device Emergency (R2D Emergency) and the research data sharing (RDS).

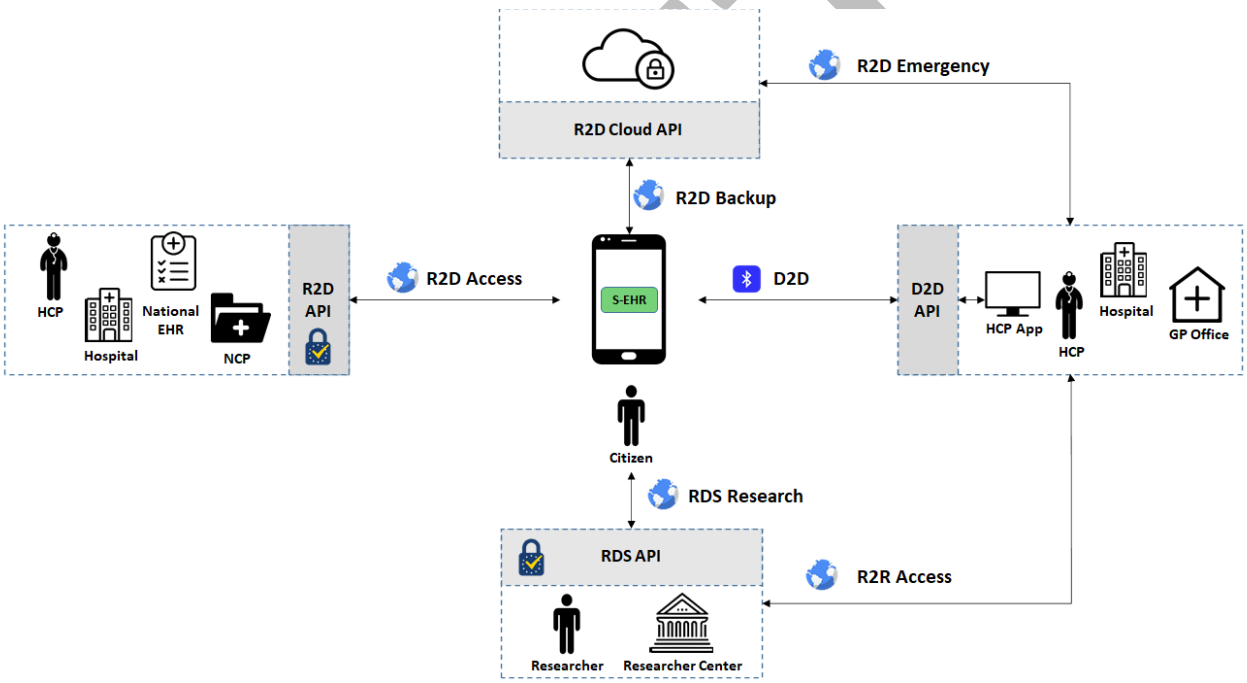



*Figure 5 – InteropEHRate protocols*

Prior to any security operation, the bootstrap phase will take place in order, for all the participants in the protocol, to agree on the necessary elements and acquire the needed Certificates as the necessary step to all the Public Key Infrastructure (PKI) frameworks. This prerequisite phase applies to all the scenarios if certificates are missing and an Internet connection is necessary.

Consent management (from the security perspective) and authorization is done through digitally signed consent exchanges and access control mechanisms where necessary. More specifically, in the context of D2D, after the demographic data exchange and prior to medical data exchange, the citizen needs to consent to the healthcare organisation of the HCP by digitally signing the provided consent. Consent authorizes the healthcare professional to access to citizen's medical data. In addition, R2D Access provides access only to authorized citizens to retrieve their own data everywhere in Europe using their unique eIDAS identity. In R2D Backup and Emergency protocols, three consents are needed. One mandatory, for

authorizing the S-EHR Cloud to store the citizen's encrypted data and two optional for authorizing healthcare professionals to access, during an emergency, the citizen's data from the S-EHR Cloud and for authorizing to access citizen's specific DICOM images not stored on the S-EHR Cloud but stored directly by the producer healthcare organizations. All consents are provided in the R2D Backup protocol, however only the mandatory consent is for the Backup scenario. The two optional consents are essential for the Emergency scenario. In addition, an attribute-based access control mechanism (ABAC) is specified for authorization of healthcare professionals in the Emergency situations. Two verification steps are specified. First the signature validity of the healthcare professional Certificate and then the allowance based on the ABAC decision. The ABAC decision is based on several defined attributes (extracted from the Certificate) and policies constructed with these attributes. In the RDS scenario a double-signed consent is specified for participation of a citizen to a research study and authorizing a selected research center to process anonymised citizen's data for research purposes. In addition, another indirect signed consent is specified for authorizing a research center to access citizen's medical data, not downloadable to the S-EHR app, directly from the Citizen's hospital. Last but not least, a private blockchain-based scheme, as an optional service, for the logging of data sharing transactions is also defined.

## 3.1 D2D Security Architecture and Models

The D2D protocol defines the set of operations that allow the exchange of health data between a S-EHR app and an HCP app in short-range distance over Bluetooth, without the usage of Internet connection **[D4.3]**. This section describes the security models in the context of D2D. The bootstrap steps annotated in Figure 6, regarding the key-pair generation and the certificate acquisition, will be described in the context of D2D and not be described again in the rest of the protocols since they are the same. Last but not least, the reader can also refer to section 3.11 of **[D3.6]** that summarises all the security common remote APIs including the interaction with the CA in order to retrieve the necessary certificates, certificate chain and validate a certificate.

Even though the interfaces are not depicted in the security models we refer to them for easier reference on the architecture of the reader. The name of the interface that is offered to the HCP app regarding the D2D protocol is named D2D. This interface contains the operations for letting the HCP app to perform tasks related to the S-EHR app, by invoking these operations, while the D2DServerSecurity and the D2DClientSecurity interfaces contain the operations for letting the HCP app and the S-EHR app establish a secure Bluetooth Connection **[D2.6]**. Both APIs will be used by the S-EHR and HCP app for security purposes. The reader can also refer to Figure 13 of **[D4.3]** for more information.

In the D2D protocol, two variants for identification were introduced in the first version of the deliverable. The identification in the first variant is done with the paper-based ID-Card of the citizen, to be coherent with the current procedures in health care institutes and a QR code generated by the hospital that includes software signatures of the HCP. The second variant, which will be used in the future, uses hardware-based signatures (Qualified) from both parties for legal binding, while the demonstration of the paper-based ID-card is omitted. This second variant, specified for experimentation reasons and not for demonstration purposes during the duration of the project, replaces handwritten signatures with electronic signatures having legal binding under the eIDAS regulation. After the successful identification process, the HCP requests the citizen's consent to access his/her data stored in the S-EHR app. Upon citizen's acceptance, a digitally signed consent is forwarded to the HCP. After this step the HCP is temporarily authorized to download/upload citizen's data from/to the S-EHR app.

The security models defined below will be the same independent of the software or hardware-based certificates and signatures. In the D2D protocol, we have two principals the $S-EHR\ App$ and the $HCP\ App$.Both principals generate a private/public key pair $r_A, t_A$and $r_B, t_B$and request from the CA to issue their Certificate. In order for the CA to generate the Certificates, both parties share their public keys. Each issued Certificate (in our case the X.509) contains information regarding the identity of each party, the corresponding public keys $t$ , while it is digitally signed by the CA's Certificate (i.e.$C_{CA}$). Each party can verify the Certificate signature (when it is necessary) with the CA's Certificate. In the following model we omit the details of the encryption and identity management crypto operations, since they are part of deliverables **[D3.6]** and **[D3.4],** respectively. Last but not least Appendix A includes the notations used for the security model.

As it is already known in this scenario, in order for the Bluetooth connection to be established, the HCP has to scan the QR code with the connection details. This QR code also includes the signature of the message (i.e.$\sigma_B$). This, apart from the integrity and authenticity, is used for identification in a latter step of the protocol. As mentioned before, the citizen's identification in the first variant will be achieved upon showing his/her real ID-card. This step is needed for legal purposes. For the second variant, where we assume qualified certificates and hardware-based signatures this first step with the ID-card is not necessary. The rest of the crypto operations are completely the same. The verification of the identity is done by the corresponding signature verification (i.e. $Ver$function), while the Nonce in the message (i.e. $N$ ) is used for freshness and replay attack avoidance. After the successful identification of the citizen, the consent (i.e. $Con$) is requested from the S-EHR app. In addition, the HCP App increases the Nonce by value k (i.e. $N+k$ ) as part of the challenge–response protocol to ensure that every challenge-response sequence is unique and to be more secure against replay attacks. The same message also includes the HCP's certificate for his/her identification. Finally, the S-EHR App signs the requested consent and sends it back to the HCP. The HCP verifies citizen's signature (i.e. $Ver$function) to validate the consent authenticity and integrity. After this final step, the HCP is authorized to download/upload citizen's medical data as long as the Bluetooth connection is established and open. Figure 6 below depicts the described security model and steps.

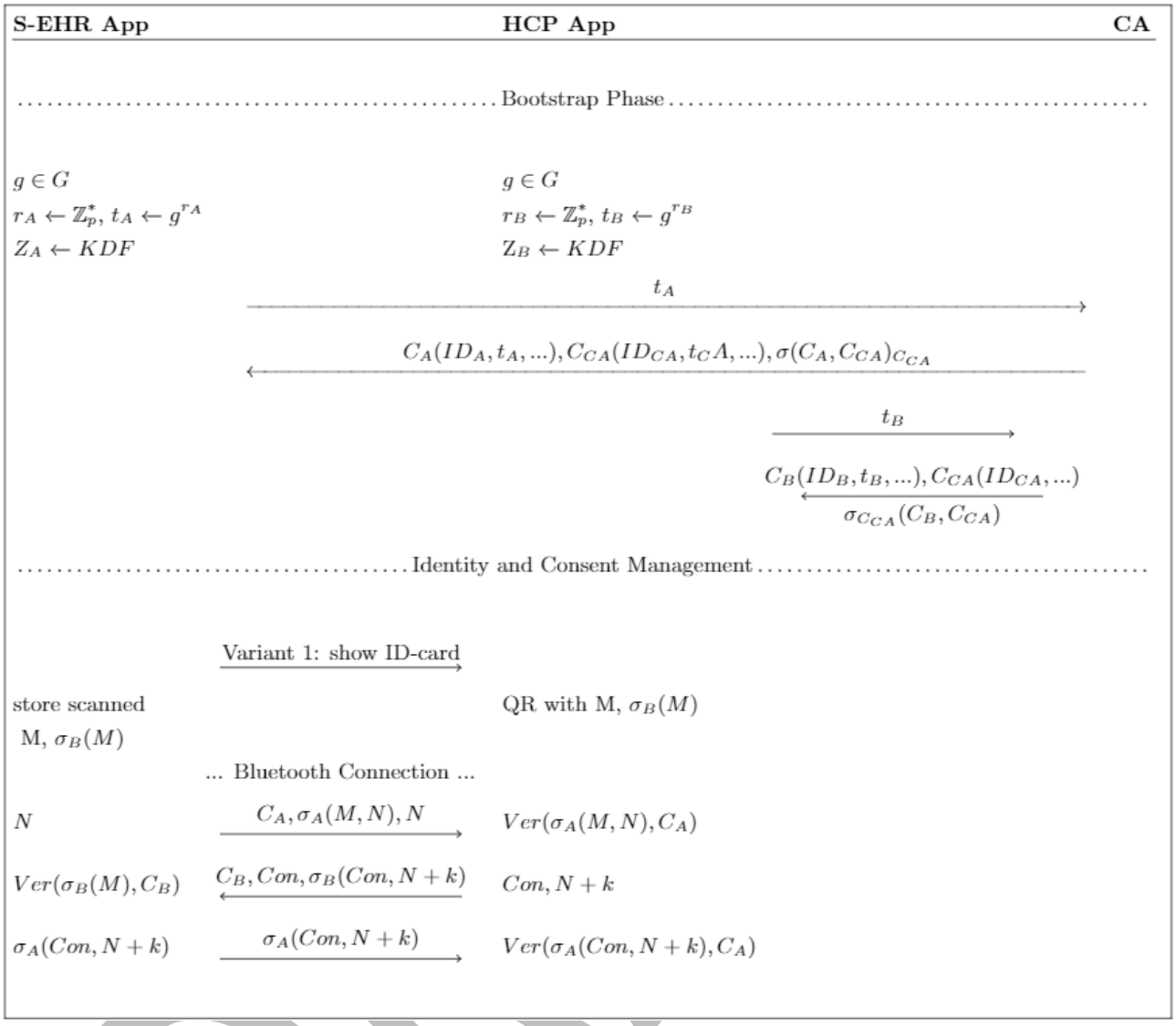



*Figure 6 – D2D crypto model*

The sequence diagram that provides a high-level overview of the D2D consent management and authorization is depicted in Figure 7, where an HCP is authorized to download/upload data stored in the S-EHR app. In the sequence diagram we omit the steps before and after as part of other deliverables.

- **Step 1**: In this step, the exchange of consent happens from the HCP app to the S-EHR app to access the S-EHR app's data.
- **Step 2**: In the case that the consent is approved, it is digitally signed from the S-EHR app and provided back to the HCP app. After this step the HCP is authorized to download/upload data stored in the S-EHR app.

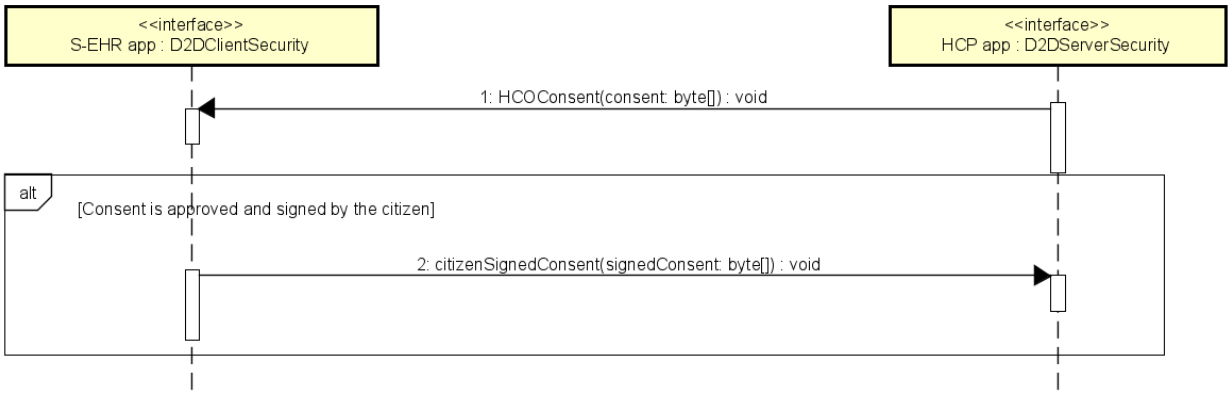



*Figure 7 – D2D sequence diagram*

## 3.2 D2D Security APIs

This section includes the (Bluetooth) remote APIs for the consent management in the D2D protocol. More specifically, when the identification process is being successfully performed, the HCP Web App initiates the procedure to request the consent for accessing the citizen's data and the S-EHR App upon accepting the consent provides the signed consent to the HCP. The security interfaces are the D2DServerSecurity and the D2DClientSecurity that contain all the security-related operations.

### 3.2.1 S-EHR App Security APIs

This section includes the interfaces that are offered to the HCP app regarding the D2D protocol, containing the operations for letting the HCP app to perform tasks related to the S-EHR app, by invoking these operations. Appendix A also includes the structure of the exchanged Bluetooth messages.

**Operation HCOConsent**

| | |
|---|---|
| **Name** | **HCOConsent** |
| **Description** | A temporary consent request from the HCP App to the S-EHR app in the form of an object, requesting access to data stored in the S-EHR app. |
| **Arguments** | • byte[] consent |
| **Return Value** | • void |
| **Exceptions** | • Security exceptions related to the Bluetooth connection<br>• Network exceptions related to Bluetooth connection failure |
| **Preconditions** | • The Bluetooth connection exists in the mobile phone that includes the S-EHR app<br>• The smart mobile device is enabled with Bluetooth v4.0 and above<br>• The session is still valid |

### 3.2.2. HCP App Security APIs

This section includes the interfaces that are offered to the S-EHR app regarding the D2D protocol, by invoking these operations. Appendix A also includes the structure of the exchanged Bluetooth messages.

**Operation citizenSignedConsent**

| Name | citizenSignedConsent |
|---|---|
| Description | The S-EHR App sends the signed consent of its user to the HCP in the case that the temporary consent has been successfully accepted. |
| Arguments | • byte[] signedConsent |
| Return Value | • void |
| Exceptions | • Security exceptions related to the Bluetooth connection<br>• Network exceptions related to Bluetooth connection failure |
| Preconditions | • The Bluetooth connection exists in the mobile phone that includes the S-EHR app<br>• The smart mobile device is enabled with Bluetooth v4.0 and above<br>• The session is still valid<br>• Successful access to the private key |

## 3.3 R2D Access Security Architecture and Models

The R2D Access protocol is an internet-based protocol, used for importing health records stored within an EHR of a healthcare organization to a smart mobile device **[D4.3]**. R2D Access leverages an eIDAS-based architecture for cross-border authentication of the citizen supporting the trust services and electronic identification, as defined by the current eIDAS framework. As already written, prior to any security operation a bootstrap phase is necessary in order for all the participants in the protocol to acquire the necessary elements. In the R2D Access, the involved parties have to acquire the necessary Certificates in order to verify the integrity and authenticity of the eIDAS assertion response.

According to the eIDAS specification, the eIDAS nodes may exchange only a restricted set of personal attributes, named eIDAS minimum data set (MDS) for natural persons, containing the person's current family name(s), the current first name(s), the date and place of birth, an eIDAS unique identifier, the current address, and the gender of a person. At least these attributes are necessary for citizen authentication/authorization. The eIDAS interoperability framework allows authentication through the exchange of SAML 2.0 messages, including personal and technical attributes. The S-EHR app requests (e.g. $authenticationRequest$) to be authenticated and hence authorization to download his/her health records for the healthcare organization. After successful eIDAS-based authentication the SAML Response $SAMLRe$(the token) is provided to the S-EHR app. The SAML Response is digitally signed (e.g. $\sigma_E(SAMLRe)$) by the eIDAS Node in order for the Healthcare organisation (of country B) to be able to check the validity of the authentication response. Both the S-EHR App and the Healthcare organization verify $Ver$ the validity of the signature as a proof that a trusted eIDAS node has generated and signed the token. The S-EHR App is authenticated $Auth$after a successful verification and hence authorized to download his/her medical data. Figure 8 depicts the R2D Access crypto model.

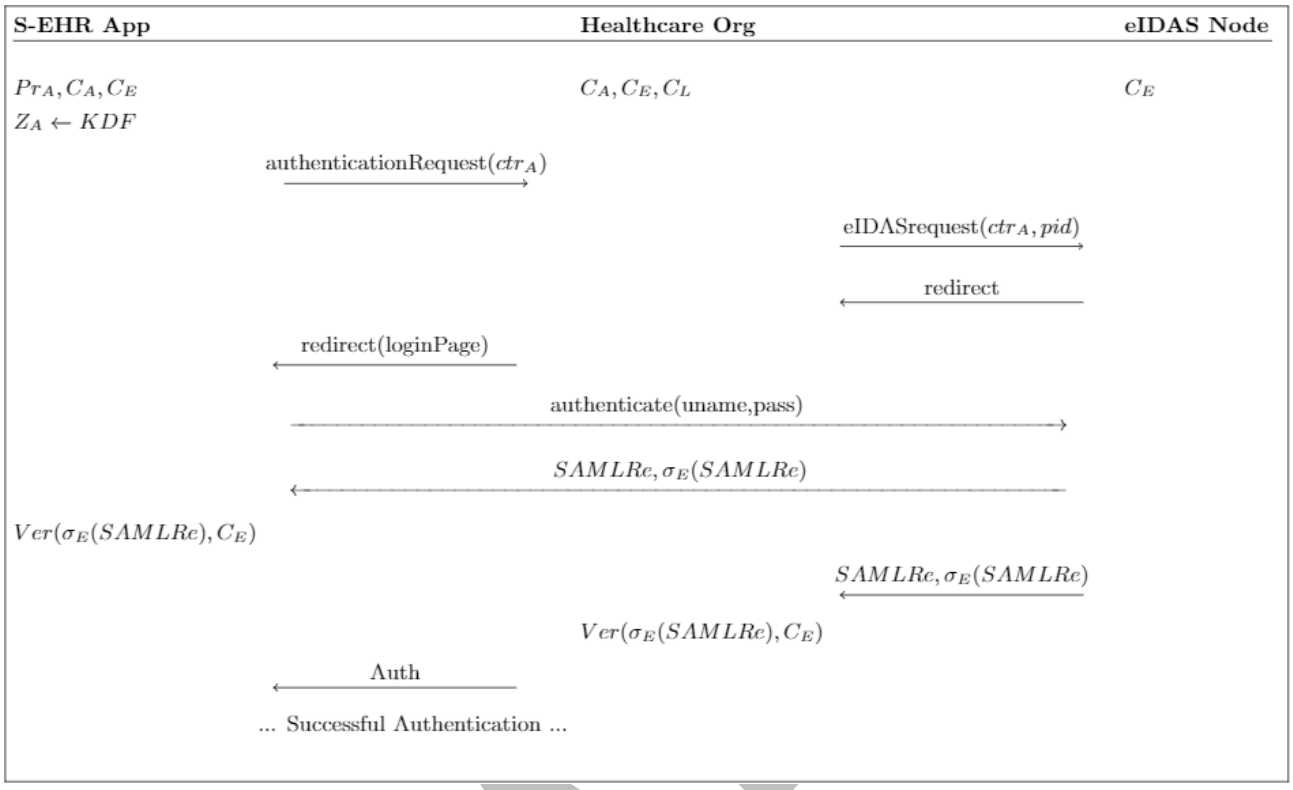



*Figure 8 – R2D Access crypto-model*

The sequence diagram that provides a high-level overview of the R2D Access authentication and authorization is depicted in Figure 9, where the S-EHR app is authorized to download his/her medical data stored in any healthcare organisation. The healthcare organisation may be a healthcare provider (e.g. a hospital, a private laboratory or a general practitioner), or a national EHR provider. In the sequence diagram we focus only on the authentication/authorization APIs and the steps before and after are omitted. The name of the interface that is offered to the S-EHR app regarding the R2D Access protocol is the R2D Access. This interface contains the operations for letting the S-EHR app to access at distance (by means of the Internet) the health data of the Citizen produced by the organisation, while R2DAccessIdentification and R2DAccessDICOM interfaces contain the operations for letting the citizen to authenticate through eIDAS and access any DICOM study of the citizen. The R2DAccessIdentification remote interface is offered by the Healthcare organisation to support the R2D Access protocol. It allows the S-EHR App to trigger the eIDAS authentication of the Citizen identity required by R2DAccess. The R2DAccessDICOM is an optional remote interface compliant to **[WADO-RS]** specification that requires the eIDAS token of the citizen for the client authentication. It allows the S-EHR App to access any DICOM study of the subject citizen referred by FHIR resources imported by means of R2DAccess or D2D. More specifically the interface R2DIdentification allows to log the citizen by means of eIDAS and the optional interface R2DAccessDICOM allows to download any DICOM study (e.g., images, signals) that are referred by the health data, but are not embedded in those data. More details are also included in the **[D3.4]**. Following a detailed description of the sequence diagram of R2D Access authentication/authorization:

- S-EHR App requests authentication prior to downloading his/her medical data. In cases where the S-EHR app is an eIDAS registered, R2D Access Server verifies the identity of the S-EHR app through the eIDAS infrastructure. Upon successful authorization the eIDAS token is returned back to the S-EHR App. This token is used in the rest of the HTTP requests as an authorization header.

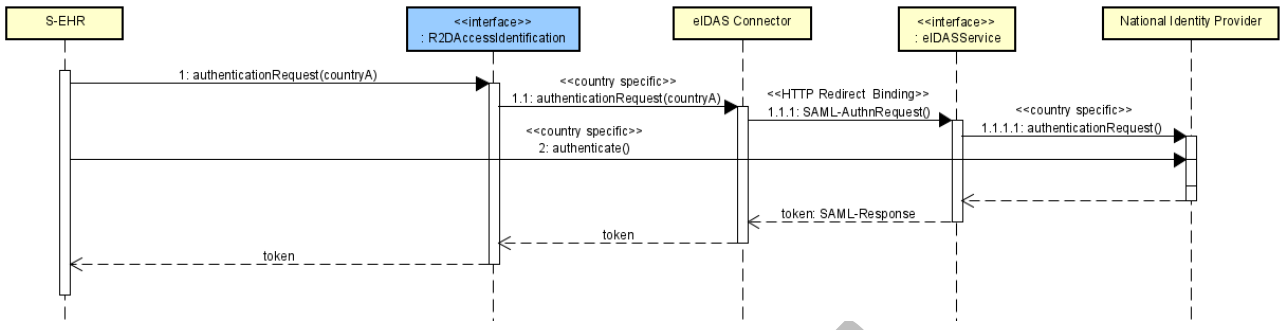



*Figure 9 – R2D Access sequence diagram*

## 3.4 R2D Access Security APIs

**Operation authenticationRequest**

| | |
|---|---|
| **Name** | authenticationRequest |
| **Description** | This operation is used by the S-EHR to start the eIDAS authentication process Operation is invoked by the S-EHR App. It is a GET request in the R2D Access endpoint http://[base url]/authenticationRequest |
| **Arguments** | HTTP Headers:<br>● Content-Type: application/json<br><br>URL params:<br>● String country: The country of the user. |
| **Return Value** | ● void<br><br>HTTP Return Codes<br>● **200 Successful:** request was successfully processed.<br>● **400 Bad request:** request could not be processed.<br>● **401 Not Authorized**: authorization is required for the interaction that was attempted.<br>● **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |
| **Exceptions** | ● Exception |
| **Preconditions** | ● N/A |

## 3.5 R2D Backup and Emergency Security Architecture and Models

The R2D Backup protocol defines the set of operations used for enabling the backup and the restore from a remote cloud of the health data repository stored locally on the mobile phone. Backup is very important to

give the citizen the option to store his/her data on a remote cloud, and restore it in case the mobile phone gets damaged, lost, broken or stolen. The citizen has the option to decide using the optional S-EHR Cloud service and backup his/her data in encrypted form. Through his/her S-EHR app, the citizen chooses their preferred S-EHR Cloud provider, creates an account and login to the services **[D6.7] [D4.3]**. Regarding the consent management and prior to the emergency the citizen needs to agree on the consent to store the data and optionally on the consents to share the citizen's health data with authorized HCPs and to share with other Healthcare Organizations the DICOM images. The optional consents will be used for authorization purposes in emergency situations. In addition, the R2D Emergency protocol defines the set of operations that allow authorized HCPs to access the encrypted health data that is backed up on a S-EHR Cloud of a citizen in need during an emergency situation over the Internet. More specifically, the HCP performs a request to the citizen's selected S-EHR Cloud provider in order to access the citizen's health data. Two verification steps are specified, first the signature validity of the healthcare professional Certificate and then the authorization mechanism allowance based on the citizen's emergency access token, along with the Healthcare Institution's attributes (including the consent), and the HCP's personal identification information. The S-EHR Cloud, upon the receiving of the HCP's request, may either approve or decline it. As a first step the S-EHR Cloud may decline any incoming request that is not coming from a trusted healthcare institution (i.e. trusted by the S-EHR Cloud provider). If the incoming request is coming from a known healthcare institution, the S-EHR Cloud contacts a trusted authorization service hosted by the S-EHR Cloud, named HCP Attributes Evaluation service, in order to evaluate the attributes that were sent with the request in the first place. This trusted service may be an Attribute-Based Access Control (ABAC) Engine. If the evaluation is successful the S-EHR Cloud creates a temporary account that may be used by the HCP's of that specific Healthcare Institution and can be used to access the citizen's health data that is stored in the S-EHR Cloud, as well as uploading health data to the S-EHR Cloud once the emergency is over. If the evaluation is not successful, the request to access the S-EHR Cloud is rejected. Apart from this authorization there is also the option for the HCP of the Healthcare Organization to perform requests to a WADO-RS server of the Healthcare Organization that contains DICOM Images of this specific citizen, to download a Study, a Series of a Study of a specific Instance of a Series.

The R2D Backup scenario requires all the involved parties to have acquired the necessary Certificates from the CA (i.e. $C_A, C_C$ ) in order to verify the integrity and authenticity of the consents. Figure 10 depicts the R2D Backup crypto model. In the context of R2D Backup protocol, the S-EHR App needs to be authenticated first with a username and password to the S-EHR Cloud ($login(u, p)$). Upon successful authentication, the S-EHR Cloud (S-EHR-C) provides a signed JWT token for future usage between S-EHR App and S-EHR Cloud communication. This token will be incorporated inside the R2D messages independently from the security library. In addition, three consents are required for the proper function of the R2D Backup and R2D Emergency protocols. The consents for S-EHR Cloud data storage ($Con_{store}$), for sharing the S-EHR Cloud stored data with authorized HCPs($Con_{share}$)and for authorized HCPs to access DICOM images stored by the producer Healthcare Organizations ($Con_{accessHOs}$) are signed from the S-EHR app and sent back to the S-EHR Cloud. Nonce (i.e. $N$ ) is used for freshness in all the consent related messages. The S-EHR App increases the Nonce by value k (i.e. $N + k$ ) as part of the challenge–response protocol to ensure that every challenge-response sequence is unique and to be more secure against replay attacks. Finally, S-EHR Cloud verifies (i.e. $Ver$function) the signatures, with the S-EHR app certificate $C_A$.

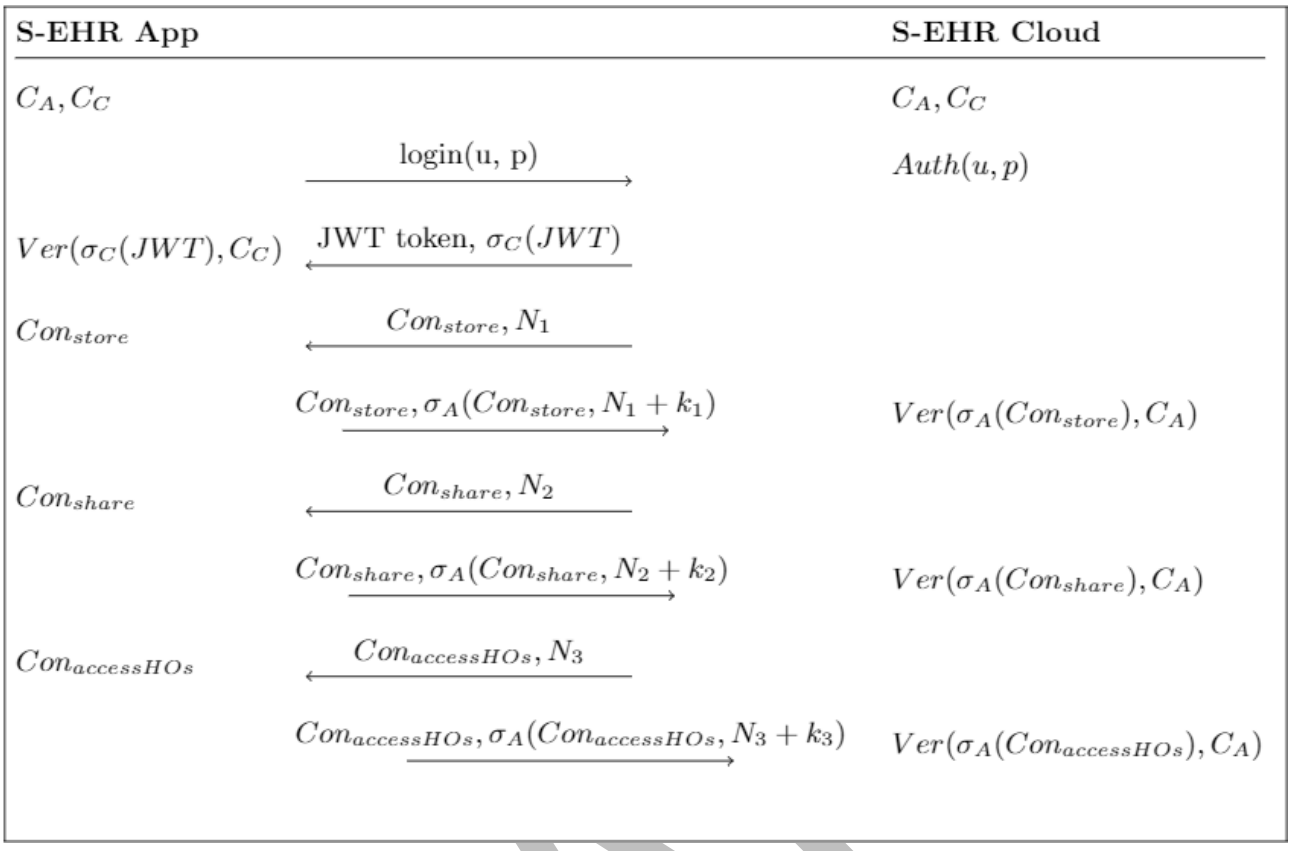



*Figure 10 – R2D Backup crypto-model*

The R2D Emergency scenario requires that only HCPs from authorised Healthcare Organizations are allowed to access a citizen's health data. Figure 11 and Figure 12 depict the R2D Emergency crypto models. In the context of R2D Emergency protocol, HCP App acquires the symmetric key, verifying its authenticity and integrity after scanning the QR code provided by the citizen (generated in R2D Backup protocol). This step is not depicted in the following models, since they are part of the **[D3.4]**. In addition, the HCP App needs to first authenticate itself to the S-EHR Cloud with a Certificate (i.e.$C_A$) issued by a CA and containing custom attributes, including the HO and the profession, the consent for sharing his/her data and the emergency token acquired from the QR code scan (i.e.$requestaccess(C_B, Con_{Share}, token_E)$). These attributes will be extracted (i.e.Ext) from the Certificate in order to be used for authorization purposes. This account should apply to the Health Organisation (HO) and not to a specific HCP according to user requirements since any qualified HCP for the HO should be able to access the S-EHR Cloud. This request access is intercepted by an Attribute-based Access Control (ABAC) engine embedded in the S-EHR Cloud, namely Attributes Evaluation service, necessary for access control purposes. Upon successful authorization, the S-EHR Cloud provides a signed JWT token, for future usage between HCP App and S-EHR Cloud communication. This token will be incorporated inside the R2D messages independently from the security library. After the successful authorization, the HCP App can download the stored encrypted data and decrypt them using the scanned symmetric key. From the decrypted data, the HCP has also access to the WADO-RS token $token_W$and the corresponding consent to share with other Healthcare Organizations the DICOM images $Con_{ShareHO}$. Both are necessary for HCP's authorization to the Healthcare Organization that store his/her DICOM images (i.e.$requestaccess(token_w, Con_{ShareHO})$ that are not able to be downloaded in the S-EHR App and hence to be backed up in the S-EHR Cloud ).

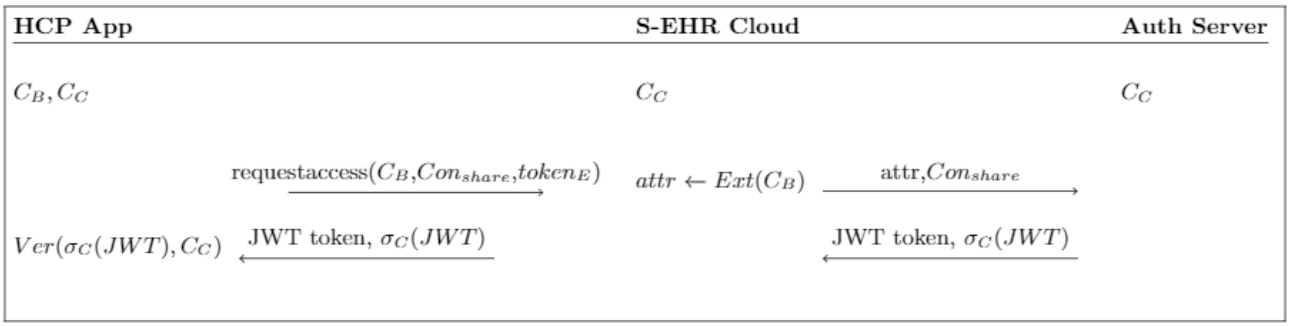



*Figure 11 – R2D Emergency crypto-model (a)*

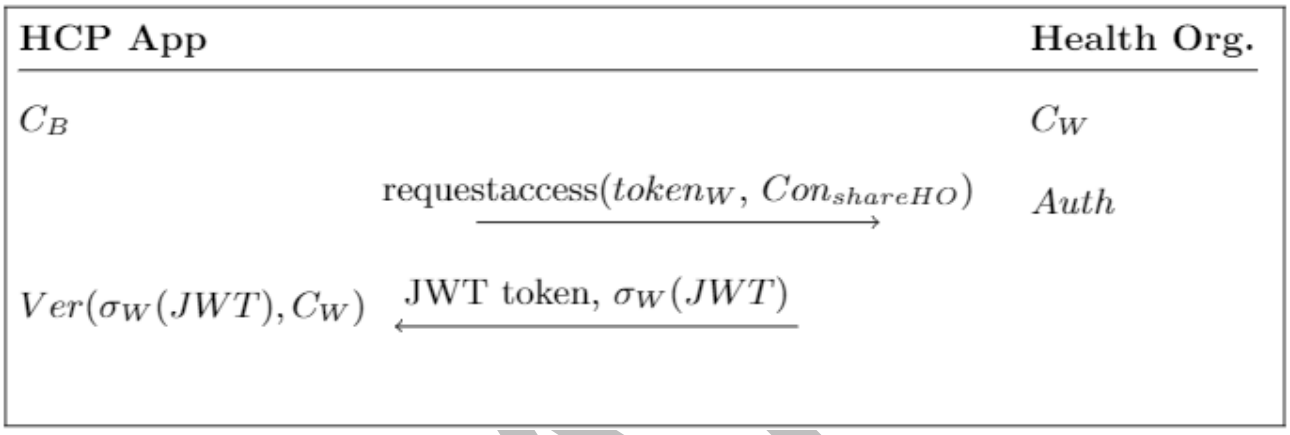


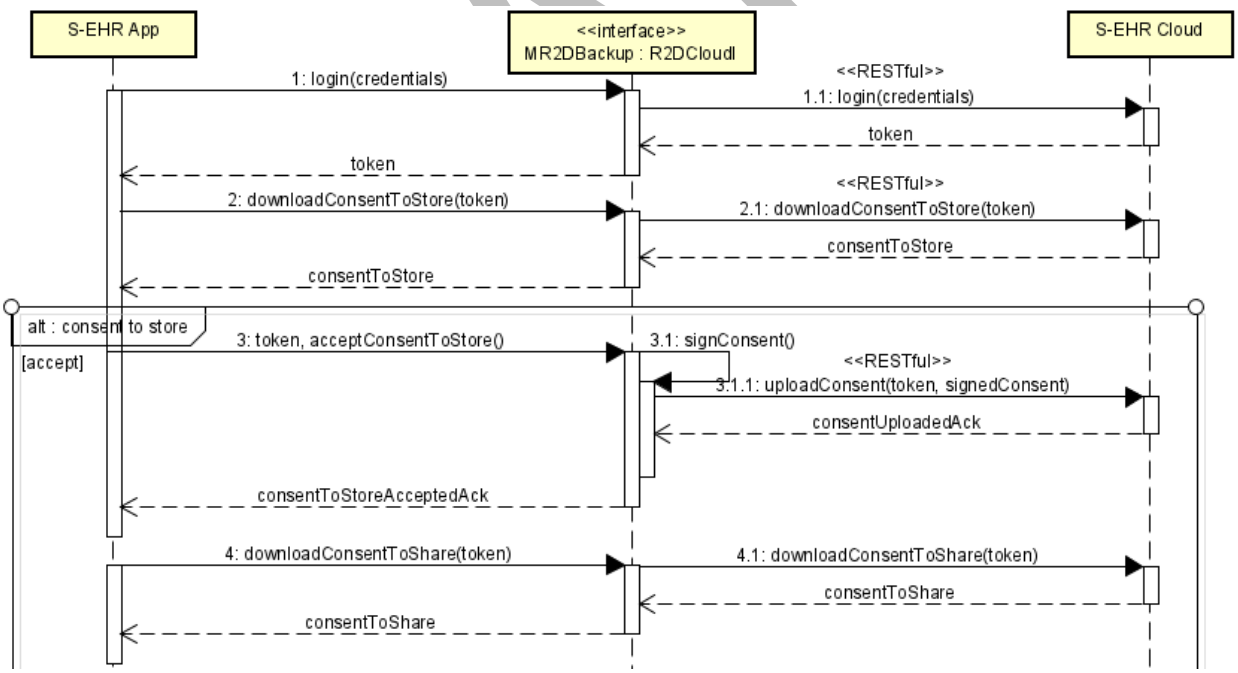

*Figure 12 – R2D Emergency crypto-model (b)*

The conceptual sequence diagram that provides a high-level overview of the R2D Backup is presented in Figure 13. Following a detailed description of the sequence diagram of R2D Backup consent management steps:

- **Step 1**: Through their S-EHR app, the citizen login to the service, by providing his/her credentials (e.g. username/password). However, the S-EHR Cloud authentication mechanism depends on the cloud provider.
- **Steps 2-3**: The first consent concerns the ability of the S-EHR Cloud provider to **store the citizen's encrypted health data**. As soon as this consent is accepted by the citizen, the health data that is stored on the citizen's smartphone is encrypted using a symmetric key created at that time (the symmetric key is stored only on the citizen's smartphone and not on the S-EHR Cloud), and uploaded on the S-EHR Cloud. This consent is retrieved by the S-EHR app, and upon citizen's acceptance is digitally signed. Afterwards, the consent including the citizen's digital signature is sent back to the S-EHR Cloud. In the case where the citizen declines this consent, he/she is not able to perform any requests to the S-EHR Cloud. This consent is given and used later in the R2D Backup protocol.
- **Steps 4-6**: The second consent regards the ability of the S-EHR Cloud provider to **share the citizen's health data with authorized HCPs** if an emergency (similar to the one described in the scenario) occurs. As soon as this consent is accepted by the citizen, a QR-code is created that the citizen should print and hold with him/her at all times. This QR-code contains information that will allow an authorized HCP to access the S-EHR Cloud that the citizen is using when needed. In addition, the QR-code contains additional information that is used by the HCP App during an emergency in order to decrypt the health data that is downloaded from the S-EHR Cloud, and encrypt new health data

related to the emergency prior to its upload on the S-EHR Cloud. In contrast with the first consent, this is not a mandatory consent. If the citizen decides to decline this consent, it means that he/she has chosen to utilize the S-EHR Cloud only as a backup service. However, if the citizen initially declines this consent, he/she also has the ability to accept it at a later time. This consent is given in the R2D Backup protocol and used later in the R2D Emergency protocol as an authorization attribute.

- **Steps 7-8**: The third consent allows the provider Healthcare Organization to **share with other Healthcare Organizations, for emergency,** by means of R2DAccessDICOM interface**, any previously created DICOM image or generated during the emergency**. What is important to note is that since there might exist several Healthcare Organizations that contain citizen's DICOM Medical Images, multiple tuples of such tokens and consents are stored in the S-EHR Cloud. When the emergency occurs the HCP of the Healthcare Organization that the citizen is taken to, performs a request to the S-EHR Cloud to obtain the above-mentioned information. With the use of this information, the HCP can perform requests to a WADO-RS server of the Healthcare Organization that contains DICOM Images of this specific citizen, to download a Study, a Series of a Study of a specific Instance of a Series. This consent is given in the R2D Backup protocol and used later in the R2D Emergency protocol as an authorization attribute. Finally, steps 9 and 10 demonstrate the cases where the citizen decides to decline each consent.



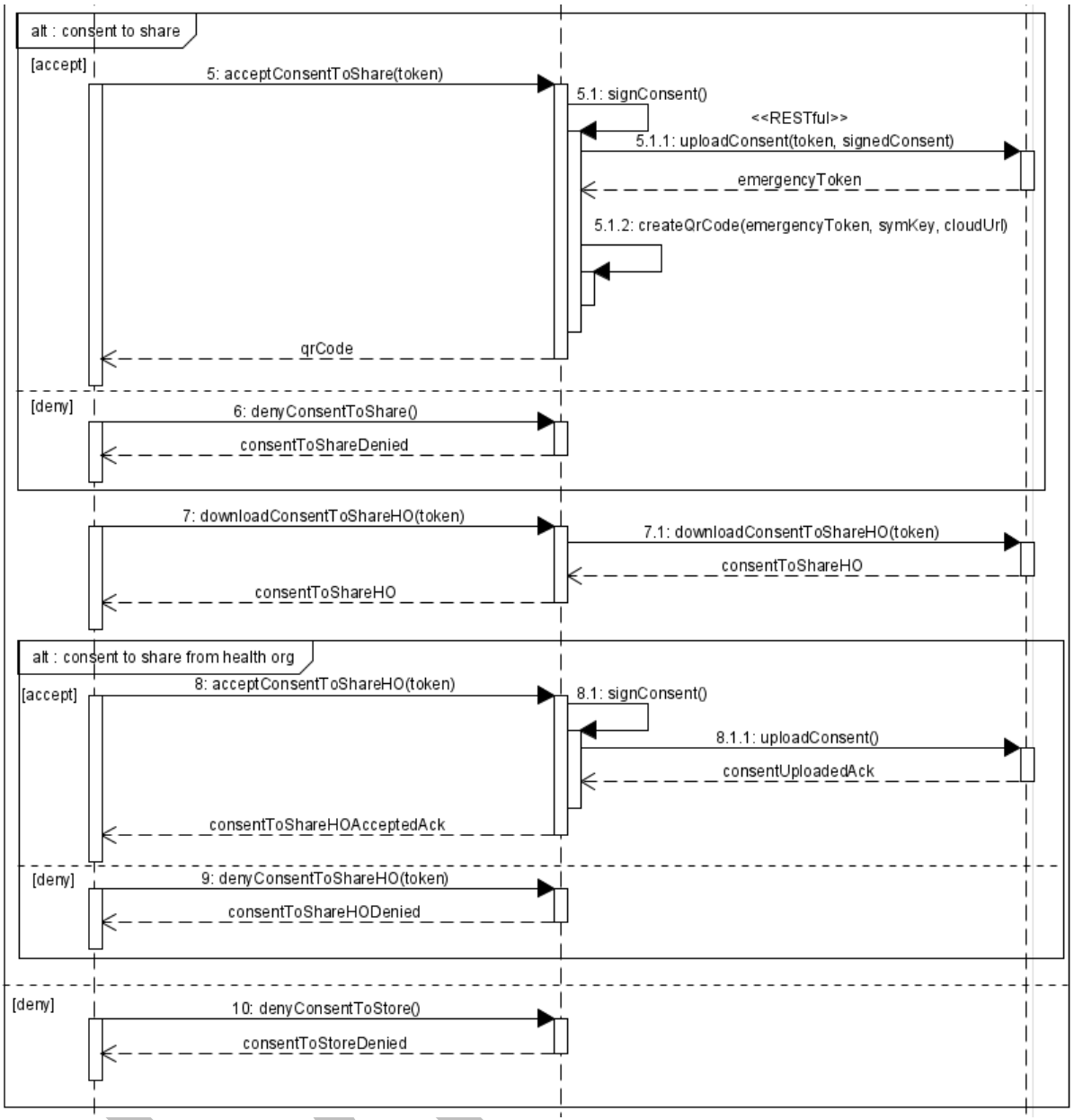



*Figure 13 – R2D Backup sequence diagram*

The conceptual sequence diagram that provides a high-level overview of the R2D Emergency is presented in Figure 14. Following a detailed description of the sequence diagram of R2D Emergency authorization steps:

- **Step 1**: The HCP should scan the QR-code found on the citizen in order to obtain the HCP authorization token, and the S-EHR Cloud URI information.
- **Steps 2-3**: The HCP requests for an access to the data stored on the S-EHR Cloud for emergency purposes. This operation allows an HCP from a specific Health Organisation to send a request to access the citizen's bucket during an emergency. A temporary account is created for the health care institution that is providing care for the patient **[D4.3]**. This account may be used by the HCPs of that specific institution during the emergency in order to download the citizen's health data, and/or upload new health data. The request includes the authorization attributes (extracted from the Certificate including the consent) and the emergency token. These attributes will be utilised by

the Attributes Evaluation service, an ABAC service provided by the S-EHR Cloud. Finally, the Attributes Evaluation service responds if the access is authorized is not.

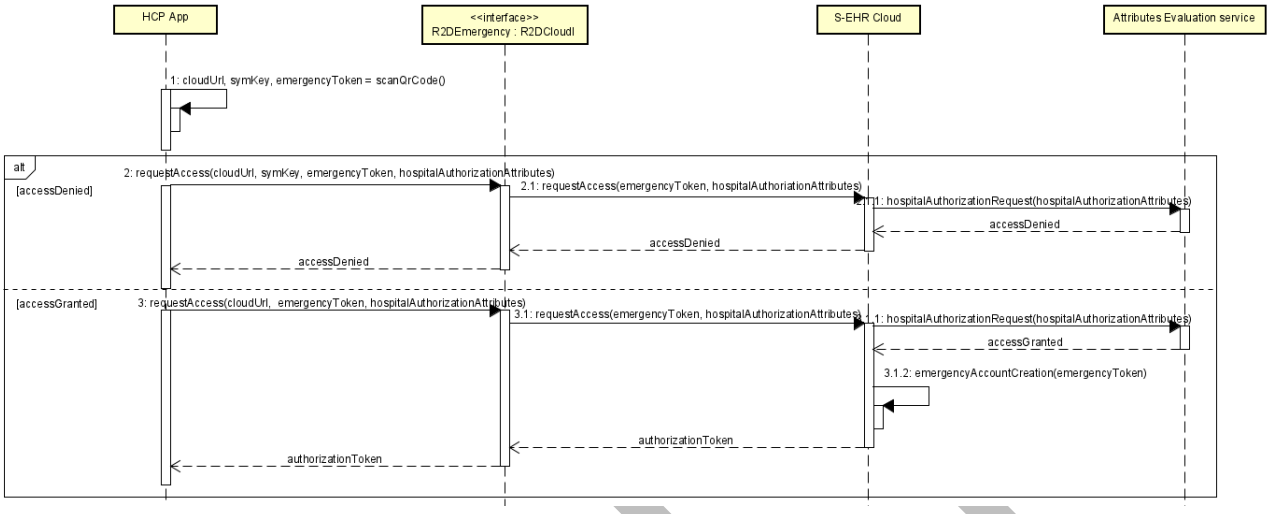



*Figure 14 – R2D Emergency sequence diagram*

## 3.6 R2D Backup and Emergency Security APIs

This section includes the remote APIs for the consent management and authorization in the D2D Backup and R2D Emergency protocols. The R2D Cloud exposes interfaces for uploading the signed consents and for authorization. In addition, the Healthcare Organizations expose interfaces for authorization prior to data sharing.

### 3.6.1 S-EHR Cloud Security APIs

**Operation download consent to store**

| | |
|---|---|
| **Name** | **download consent to store** |
| **Description** | Download unsigned consent to store health data to the S-EHR Cloud. It is a GET request in the S-EHR Cloud endpoint http://[base url]/citizen/store |
| **Arguments/Headers** | • HTTP Headers:<br>  ○ “Authorization”: Authentication token - JSON Web token |
| **Return Value** | • Consent to share health data that is stored on the S-EHR-C with authorized HCPs: String<br><br>HTTP Return Codes<br>• **200 Successful:** request was successfully processed.<br>• **400 Bad request:** request could not be processed.<br>• **401 Not Authorized**: authorization is required for the interaction that |

| | |
|---|---|
| | was attempted.<br>• **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |
| **Exceptions** | • Missing header: Authentication token |
| **Preconditions** | • The citizen should login to the S-EHR Cloud in order to obtain the authentication token. |

**Operation upload consent to store**

| | |
|---|---|
| **Name** | **upload consent to store** |
| **Description** | Upload the signed consent to store health data on the S-EHR Cloud. The citizen may upload encrypted health data to their bucket on the S-EHR Cloud only after this consent's upload. It is a POST request in the S-EHR Cloud endpoint http://[base url]/citizen/upload/store |
| **Arguments/Headers** | • HTTP Headers:<br>  ○ "Authorization": Authentication token - JSON Web token<br>• Signed consent: Bytes |
| **Return Value** | • Acknowledgement of the upload of the signed consent<br><br>HTTP Return Codes<br>• **200 Successful:** request was successfully processed.<br>• **400 Bad request:** request could not be processed.<br>• **401 Not Authorized**: authorization is required for the interaction that was attempted.<br>• **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |
| **Exceptions** | • Missing header: Authentication token<br>• Missing argument: Signed consent |
| **Preconditions** | • The citizen should login to the S-EHR Cloud in order to obtain the authentication token.<br>• The citizen should before download the consent from the S-EHR Cloud. |

**Operation download consent to share**

| Name | **download consent to share** |
|---|---|
| Description | Download unsigned consent to store health data to the S-EHR Cloud. It is a GET request in the S-EHR Cloud endpoint http://[base url]/citizen/share |
| Arguments/Headers | • HTTP Headers:<br>  ○ "Authorization": Authentication token - JSON Web token |
| Return Value | • Consent to share health data that is stored on the S-EHR-C with authorized HCPs: String<br><br>HTTP Return Codes<br>• **200 Successful:** request was successfully processed.<br>• **400 Bad request:** request could not be processed.<br>• **401 Not Authorized**: authorization is required for the interaction that was attempted.<br>• **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |
| Exceptions | • Missing header: Authentication token |
| Preconditions | • The citizen should login to the S-EHR Cloud in order to obtain the authentication token. |

**Operation upload consent to share**

| Name | **upload consent to share** |
|---|---|
| Description | Upload the signed consent to share health data with authorized HCPs. Authorized HCPs may gain access to the citizen's health data only if the citizen has previously agreed on this consent. It is a POST request in the S-EHR Cloud endpoint http://[base url]/citizen/upload/share |
| Arguments/Headers | • HTTP Headers:<br>  ○ "Authorization": Authentication token - JSON Web token<br>• Signed consent: Bytes |
| Return Value | • Acknowledgement of the upload of the signed consent<br><br>HTTP Return Codes<br>• **200 Successful:** request was successfully processed.<br>• **400 Bad request:** request could not be processed.<br>• **401 Not Authorized**: authorization is required for the interaction that |

| | |
|---|---|
| | was attempted.<br>• **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |
| **Exceptions** | • Missing header: Authentication token<br>• Missing argument: Signed consent |
| **Preconditions** | • The citizen should login to the S-EHR Cloud in order to obtain the authentication token.<br>• The citizen should before download the consent from the S-EHR Cloud. |

**Operation download consent to share from HO**

| | |
|---|---|
| **Name** | **download consent to share from HO** |
| **Description** | Download unsigned consent to store health data to the S-EHR Cloud. It is a GET request in the S-EHR Cloud endpoint http://[base url]/citizen/shareho |
| **Arguments/Headers** | • HTTP Headers:<br>  ○ "Authorization": Authentication token - JSON Web token |
| **Return Value** | • Consent to share health data that is stored on the S-EHR-C with authorized HCPs: String<br><br>HTTP Return Codes<br>• **200 Successful:** request was successfully processed.<br>• **400 Bad request:** request could not be processed.<br>• **401 Not Authorized**: authorization is required for the interaction that was attempted.<br>• **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |
| **Exceptions** | • Missing header: Authentication token |
| **Preconditions** | • The citizen should login to the S-EHR Cloud in order to obtain the authentication token. |

**Operation upload consent to share from HO**

| | |
|---|---|
| **Name** | **upload consent to share from HO** |

| | |
|---|---|
| **Description** | Upload the signed consent to share health data with authorized HCPs. Authorized HCPs may gain access to the citizen's health data only if the citizen has previously agreed on this consent. It is a POST request in the S-EHR Cloud endpoint http://[base url]/citizen/upload/shareho |
| **Arguments/Headers** | • HTTP Headers:<br>  ○ "Authorization": Authentication token - JSON Web token<br>• Signed consent: Bytes |
| **Return Value** | • Acknowledgement of the upload of the signed consent<br><br>HTTP Return Codes<br>• **200 Successful:** request was successfully processed.<br>• **400 Bad request:** request could not be processed.<br>• **401 Not Authorized**: authorization is required for the interaction that was attempted.<br>• **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |
| **Exceptions** | • Missing header: Authentication token<br>• Missing argument: Signed consent |
| **Preconditions** | • The citizen should login to the S-EHR Cloud in order to obtain the authentication token.<br>• The citizen should before download the consent from the S-EHR Cloud. |

**Operation withdraw consent to share**

| | |
|---|---|
| **Name** | **withdraw consent to share** |
| **Description** | Withdraw consent to share health data with authorized HCPs. Authorized HCPs may no longer gain access to the citizen's health data on the S-EHR Cloud. It is a POST request in the S-EHR Cloud endpoint http://[base url]/citizen/withdraw/share |
| **Arguments/Headers** | • HTTP Headers:<br>  ○ "Authorization": Authentication token - JSON Web token |
| **Return Value** | • Acknowledgement of the withdrawal of the consent a) The consent is revoked from the S-EHR Cloud, b) The HCP authorization token can no longer be used<br><br>HTTP Return Codes |

| | |
|---|---|
| | • **200 Successful:** request was successfully processed.<br>• **400 Bad request:** request could not be processed.<br>• **401 Not Authorized**: authorization is required for the interaction that was attempted.<br>• **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |
| **Exceptions** | • Missing header: Authentication token |
| **Preconditions** | • The citizen should login to the S-EHR-C in order to obtain the authentication token.<br>• The citizen should have already agreed to the consent. |

**Operation requestaccess**

| | |
|---|---|
| **Name** | requestaccess |
| **Description** | An HCP of a healthcare institution requests access to the citizen's health data during an emergency situation. HCP's request to access the citizen's bucket during an emergency. A temporary account is created for the health care institution that cures the patient. This temporary account may be used by the HCPs of that specific institution during the emergency in order to download the citizen's health data, and/or upload new health data in a bucket dedicated for that specific emergency. The HCP should also include the set of attributes that the S-EHR-C utilizes in order to authenticate the health care institution. It is a POST request in the S-EHR Cloud endpoint http://[base url]/hcp/requestaccess |
| **Arguments/Headers** | • HTTP Headers:<br>  ○ "Authorization": Citizen's Emergency Access token - JSON Web token<br>  ○ Health care institution attributes: String |
| **Return Value** | • Health care institution authentication token: JSON Web token<br><br>HTTP Return Codes<br>• **200 Successful:** request was successfully processed.<br>• **400 Bad request:** request could not be processed.<br>• **401 Not Authorized**: authorization is required for the interaction that was attempted.<br>• **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |

| | |
|---|---|
| **Exceptions** | • Missing header: Citizen's Emergency Access token<br>• Missing argument: Health care institution attributes<br>• Missing argument: HCP attributes |
| **Preconditions** | • The citizen should have agreed to share their health data with authorized HCPs during emergency situations.<br>• The HCP should scan the QR-code found on the citizen in order to obtain the HCP authorization token, and the S-EHR Cloud URI information. |

### 3.6.2 Health Organisation Security APIs

**Operation requestaccess**

| | |
|---|---|
| **Name** | requestaccess |
| **Description** | An HCP of a healthcare institution requests access to the citizen's health data during an emergency situation. HCP's request to access the hospital's WADO-RS server in an emergency. It is a GET request in the HOs endpoint http://[base url]/requestaccess |
| **Arguments/Headers** | • HTTP Headers:<br>  ○ "Authorization": Citizen's Emergency Access token - JSON Web token<br>  ○ WADO-RS Authorization token: String<br>  ○ Consent: String *(Citizens' consent to share from HO downloaded from the S-EHR Cloud during the emergency)* |
| **Return Value** | • Health care institution authentication token: JSON Web token<br><br>HTTP Return Codes<br>• **200 Successful:** request was successfully processed.<br>• **400 Bad request:** request could not be processed.<br>• **401 Not Authorized**: authorization is required for the interaction that was attempted.<br>• **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |
| **Exceptions** | • Missing header: Authentication token<br>• Missing consent: WADO-RS authorization token<br>• Missing consent: Citizen's consent |

| Preconditions | • The citizen should have agreed to share their health data with authorized HCPs during emergency situations.<br>• The citizen should have agreed to share their health stored in the hospital data with authorized HCPs during emergency situations.<br>• The HCP should scan the QR-code found on the citizen in order to obtain the HCP authorization token, and the S-EHR-C URI information. |
|---|---|

## 3.7 RDS Security Architecture and Models

The RDS protocol addresses the general problem of collecting health data for cross-border medical research in order to enable secure and privacy-preserving cross-border data collection **[D4.9]**. This includes sharing health data from both the mobile S-EHR App and any HO (e.g. an hospital) hosting data not able to be downloaded in the mobile. In this protocol the S-EHR App supports opt-in to a study and uploads anonymised data on the selected RRC (Reference Research Center). The RDS protocol also provides the ability of direct communication between a research centre and any HO, of the citizen participating in a research study, storing data that cannot be directly obtained from the Citizen's mobile device. This operation is realized through the remote-to-research access protocol (R2R-Access), very similar to the R2D-Access protocol.

To assure the patient's privacy, two variants of RDS are supported: one a) with pseudo-ids and another b) with pseudonyms. The pseudo-id will be generated by the Pseudo-identity Generation Service at each Research Centre participating in the study, whereas the pseudonym will be generated by an independent Pseudonym Provider (PP). In addition, in the second variant we assume a trust-relation is already established among the Pseudonym Provider (PP), the eIDAS Node and the identity provider (IDP) with which the S-EHR App has been registered. To achieve such a trust-relation, SAML assertions that represent security claims produced by the eIDAS Node are leveraged, including authentication, authorization, and attribute statements. The eIDAS SAML response contains two parts, both signed by the eIDAS Node: a) one which is the actual SAML response of the eIDAS Node (this represents the long-term id $L_{ID}$), and b) one anonymous SAML assertion, without any identified information (this represents the transient id $T_{ID}$ ). The first one is the one used for the S-EHR App authentication in the R2D Access protocol and in the R2R Access protocol (similar to R2D Access) and the second one is the anonymous assertion that will be used in the second variant of the RDS protocol, in order to assure that the S-EHR App is an authenticated member to the PP without disclosing an actual identity. More details regarding authentication and this trust-relation of the second variant are included in deliverable **[D3.4]**.

The description of the security models in the context of RDS for consent management and authorization between the S-EHR app and the RRC follows. In addition, the security model regarding R2R Access is omitted since it is simpler that the R2D Access, with the difference that the requesting RRC has to provide to the data provider HO the citizen's consent ($Con_{HO}$) to download the heath data instead of the citizen's eIDAS token. As with the previous protocols, S-EHR App and RRC already have access to the needed Certificates (e.g. $C_A, C_R, C_W$) and S-EHR App has already acquired the anonymous eIDAS SAML assertion $T_{ID}$(acquired in the context of R2D Access protocol) or any other time on demand prior to the RDS protocol. In the context of consent management, the RDS protocol involves two consents: a) a consent $Con$ for participation of a citizen to a research study (**enrollment consent**) and b) an indirect consent $Con_{HO}$to the

Reference Research Centre to request anonymized health content directly from the Citizen's HO that originated the content (**direct access consent**). The second consent is necessary for the R2R Access protocol and is needed in cases where health data necessary for research is not directly downloadable from the Citizen's mobile device (such as a large media file), or it cannot be properly anonymized inside the mobile device for research purposes. Both consents are signed by the S-EHR App $\sigma_A$,while the first one double signed (signed also by the RRC $\sigma_B$).

Along with the direct access consent the S-EHR App provides a Request Authorization Token $token_W$ that testifies that the Citizen has given consent to the RRC to retrieve the objects from the hospital. The verification of the identity is done by the corresponding signature verification (i.e. $Ver$function), while the Nonce (i.e. $N$ ) is used for freshness. When data is retrieved from the S-EHR App to be sent to the RRC, unstructured data objects that need to be anonymized before being sent to the RRC are identified by the S-EHR App. The S-EHR App provides references (resource identifiers) that allow their retrieval from the originating HO. For each object, the S-EHR App also provides the *Request Authorization Token* $token_W$including the direct access consent $Con_{HO}$to the RRC to access the objects from the wado-rs server of the content originating HO. Upon reception, the RRC makes a request to the HO holding the data and the HO exposes the result for subsequent download by the RRC.

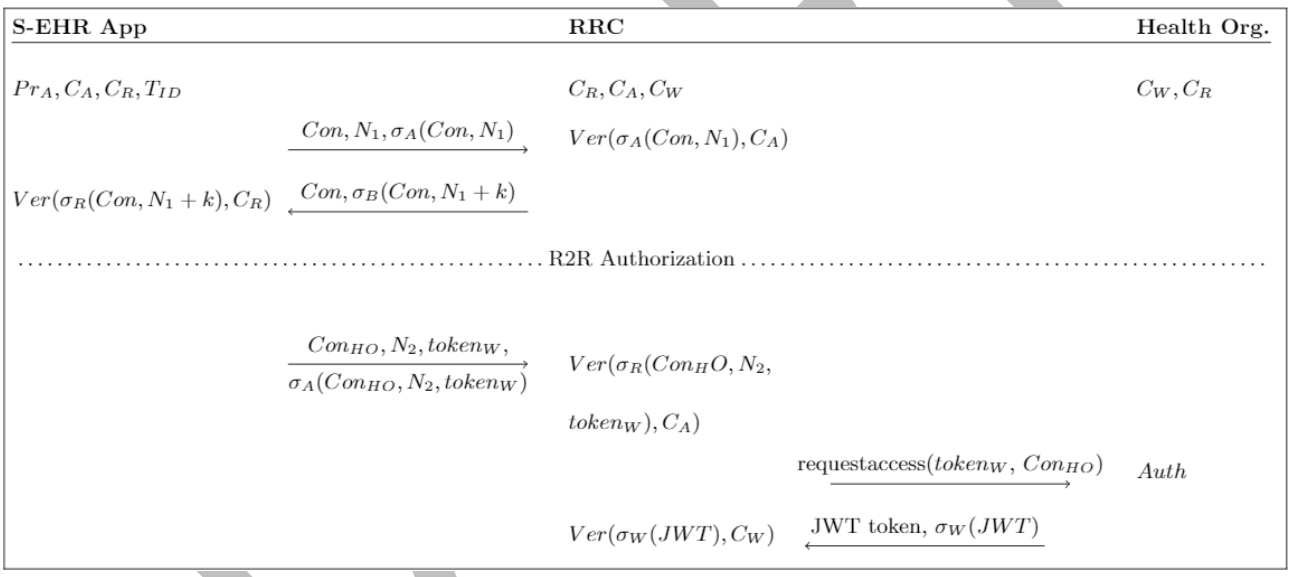



*Figure 15 – RDS crypto-model*

The conceptual sequence diagram that provides a high-level overview of RDS in Figure 16, where the consent management and authorization request flows are depicted. Following a detailed description of the sequence diagram:

- **Step 1:** This step demonstrates the consent of the Citizen to enrol into a specific research study (enrolment consent). The signed consent includes the newly generated study-specific pseudonym or pseudo-identity and the S-EHR App ID. The receiving research center checks the signature validity of the signed consent, signs and returns the contract signed by both parties (double-signed consent).
- **Step 2**: This step demonstrates an indirect consent to the Reference Research Centre to request anonymized content directly from the originating HO (direct access consent).

- **Step 3-4**: This step demonstrates the request to access anonymized content directly from the Citizen's originating HO. Both the Request Authorization Token and the consent to the RRC are mandatory to successfully authorize the RRC.


*Figure 16 – RDS sequence diagram*

In addition, the usage of blockchain/distributed ledger technology for secure logging of sharing transaction events is used as an optional service. Such logs have the property that it is virtually impossible to change the order and/or contents of the logged events, and that the logs are highly available to those that are authorized (e.g. has permission to access the blockchain inside the RRC). The S-EHR app (optionally) logs the enrolment to a research study in the blockchain as a reference for search purposes. Such transaction logs can assist researchers to easily search what type of data are shared and when, while there is the assertion that this information is correct and valid.

## 3.8 RDS Security APIs

This section includes the remote APIs for the consent management and authorization in the RDS protocol. More details can also be found in **[D4.9]**.

### 3.8.1 RRC App Security APIs

**Operation sendEnrollmentConsent**

| Name | POST sendEnrollmentConsent |
|---|---|
| Description | Send the Citizen's signed consent of enrolling into a specific study. The receiving research center checks the signature validity of the signed consent, signs and returns the contract signed by both parties. It is a POST request in the RRC endpoint http://<BASE_ADDR>/sendEnrollmentConsent?studyID=<studyID> |

| | |
|---|---|
| **Arguments/Headers** | HTTP Headers:<br>● Content-Type: application/fhir+json<br><br>URL params:<br>● studyID: the ID of the study in which the Citizen is enrolling;<br><br>The POST body content is formed by the following files:<br>● signedConsent: a digitally signed document containing the Citizen's consent to participate in the study;<br>● a json file containing the following objects:<br>○ citizenPseudo: pseudonym or pseudo-identity generated for the Citizen;<br>○ citizenCertificate: certificate of the Citizen issued by a Certification Authority and sent to the RC so that it can verify the digital signature;<br>○ enrollmentCriteriaData: encrypted data values corresponding to the enrolment criteria, so that the RC can cross-check their validity;<br>○ sehrAppId: identifier of the S-EHR App product and instance, for traceability of the S-EHR App product being used) |
| **Return Value** | The response body is composed by:<br>● SignedContract: the consent contract where the Research Centre has added its own digital signature, and which is now signed by both parties;<br>● the certificate of the Research Centre that certifies the authenticity of the digital signature.<br><br>HTTP Return Codes<br>**200 Successful**: request was successfully processed.<br>**400 Bad Request**: search could not be processed or failed basic FHIR validation rules.<br>**401 Not Authorized**: authorization is required for the interaction that was attempted.<br>**403 Forbidden**: client is not allowed to access requested resources due to security policy.<br>**404 Not Found**: resource type not supported, or not a valid FHIR endpoint.<br>**406 Not Acceptable**: client requested a not supported content-type format.<br>**500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |
| **Exceptions** | The call's exceptions returned are added as text messages within the HTTP responses. The possible exceptions returned can be:<br>● invalid content (study ID, pseudo-identity, consent form);<br>● digital signature cannot be verified;<br>● enrolment criteria not met |
| **Preconditions** | ● The S-EHR App of the Citizen must have verified the eligibility of the |

| | |
|---|---|
| | Citizen to participate in the study through checking the enrolment criteria.<br>● The S-EHR App must have access to the Citizen's private key in order to sign the consent.<br>● The S-EHR App must have generated a pseudonym or pseudo-identity to be used in the study, which conforms to the RDD of the study. |

**Operation sendHOConsent**

| | |
|---|---|
| **Name** | **POST sendHOConsent** |
| **Description** | Send the Citizen's electronically signed consent of downloading the data directly from the Citizen's hospital. The receiving RC checks the signature validity of the signed consent. It is a POST request in the RRC endpoint http://<BASE_ADDR>/sendHospitalConsent?studyID=<studyID> |
| **Arguments** | HTTP Headers:<br>● Content-Type: application/fhir+json<br><br>URL params:<br>● studyID: the ID of the study in which the Citizen is enrolling;<br><br>The POST body content is formed by the following files:<br>● signedConsent: a digitally signed document containing the Citizen's consent to share his/her data from the hosting hospital ;<br>● a json file containing the following objects:<br>○ citizenPseudo: pseudonym or pseudo-identity generated for the Citizen;<br>○ citizenCertificate: certificate of the Citizen issued by a Certification Authority and sent to the RC so that it can verify the digital signature;<br>○ enrollmentCriteriaData: encrypted data values corresponding to the enrolment criteria, so that the RC can cross-check their validity;<br>○ sehrAppId: identifier of the S-EHR App product and instance, for traceability of the S-EHR App product being used) |
| **Return Value** | HTTP Return Codes<br>**200 Successful**: request was successfully processed.<br>**400 Bad Request**: search could not be processed or failed basic FHIR validation rules.<br>**401 Not Authorized**: authorization is required for the interaction that was attempted.<br>**403 Forbidden**: client is not allowed to access requested resources due to security policy.<br>**404 Not Found**: resource type not supported, or not a valid FHIR endpoint.<br>**406 Not Acceptable**: client requested a not supported content-type format. |

| | |
|---|---|
| | **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |
| **Exceptions** | The call's exceptions returned are added as text messages within the HTTP responses. The possible exceptions returned can be:<br>● invalid content (study ID, pseudo-identity, consent form);<br>● digital signature cannot be verified;<br>● enrolment criteria not met |
| **Preconditions** | ● The S-EHR App of the Citizen must have verified the eligibility of the Citizen to participate in the study through checking the enrolment criteria.<br>● The S-EHR App must have access to the Citizen's private key in order to sign the consent.<br>● The S-EHR App must have generated a pseudonym or pseudo-identity to be used in the study, which conforms to the RDD of the study.<br>● The S-EHR App must have signed a consent of enrolling into a specific study. |

### 3.8.2 Health Organisation (R2R Access) Security APIs

**Operation requestaccess**

| | |
|---|---|
| **Name** | requestaccess |
| **Description** | The researcher of the RRC requests access to the citizen's health data for research purposes. Principal Investigator requests to access the hospital's WADO-RS server. It is a GET request in the HOs endpoint http://[base url]/requestaccess |
| **Arguments/Headers** | ● HTTP Headers:<br>○ "Authorization": Citizen's Authorization token - JSON Web token<br>○ Consent: String |
| **Return Value** | ● Health care institution authentication token: JSON Web token<br><br>HTTP Return Codes<br>● **200 Successful:** request was successfully processed.<br>● **400 Bad request:** request could not be processed.<br>● **401 Not Authorized**: authorization is required for the interaction that was attempted.<br>● **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |

| | |
|---|---|
| **Exceptions** | • Missing header: Authentication token<br>• Missing consent: WADO-RS authorization token<br>• Missing consent: Citizen's consent |
| **Preconditions** | • The citizen should have agreed to share their health data with authorized RRCs. |

## 3.9 RDS Blockchain Optional Service and APIs

This section describes InteropEHRate's optional blockchain-based service used for secure logging of the data transactions that have occurred in the context of RDS scenario. InteropEHRate will be capable of forming trustworthy and auditable research data value chains inside a research center. To do so, a blockchain infrastructure will be used to offer private DLTs to enable secure and auditable data transaction management for the research data inside the same research center. Blockchain technology, combined with lightweight cryptographic primitives, will be used for securely storing transactions of research data while keeping data integrity at the highest level of assurance (e.g. secure storage, access control etc.). The vision of such a blockchain-based service is to enable decentralized data ownership safeguarding and data traceability throughout the entire lifecycle of healthcare data; from the origin (citizens) through every point of contact that wishes to transact with a specific dataset. By using blockchain technology in tandem with cryptographic primitives (e.g., Non-Fungible Tokens (NFTs) as tokens stored also on the ledger) enables us to represent a digital asset (healthcare data) on the blockchain and certify that they are unique and, thus, interchangeable. More specifically, the value of such a blockchain service is three-fold: a) citizens will be in position to know/trace who is having access to their data (traceability, provenance tracking) by integrating smart contracts for validating transaction rules even after the reading/extraction of a specific dataset, and b) researchers will be assured that citizens will not be able to deny that they consented to sharing their data (undeniability, non-repudiation) and c) researchers will in position to extract knowledge from the stored transactions (off-chain knowledge extraction) in an efficient way.

Our architecture is based on a private Hyperledger Fabric blockchain, while each research center may have its own private blockchain to securely log citizen's data transactions. From the user perspective, private blockchains allow establishing an access control mechanism to participate (join) to the blockchain, so it is quite straightforward to control access. The InteropEHRate blockchain will inherit the intrinsic functions from current blockchain techniques to achieve the storage of transactions for all authenticated parties (e.g. researchers from RRCs) on a private ledger. Data transactions can be any process upon the anonymised data, such as sharing data to the RRC, authorized access, or analysis on the data, while each transaction is triggered automatically upon data sharing, and request for usage/access. The actors involved in the InteropEHRate blockchain-based architecture are the patient through the S-EHR app that can share his/her medical data and the researcher that can access and analyze the data for research purposes:

- **Data Providers** - authorised patients part of the InteropEHRate network that share his/her medical data in an anonymised form.
- **Data Seekers** - authorised researchers inside the research center that requests access to the data to perform a research study.

The architecture includes two smart contracts for secure and privacy preserving transaction logging as part of the workflow. Since the smart contracts are pieces of code, there should be a definition of the functions they will provide to the rest of the tools. The first step for that design is the description of those smart contracts and the interactions with other components in the general architecture. The two main processes where a smart contract is involved are one for data upload transaction and one for data download transaction. In what follows, we describe in more detail the main concepts surrounding the operation and are used as the core contents of these smart contracts. This sets the baseline of the envisioned functionalities of smart contracts, as a core enabler for data logging.

**Smart Contract for Data Upload Transaction**

- **Objective**: S-EHR app logs the transaction of his/her shared data for research purposes in an auditable manner. Upon sharing research data, a log translation is created in the blockchain that includes a timestamp, the Reason, the studyID and citizenPseudo parameters that are already defined in **[**D4.9**]**.
- **Triggered by whom**: S-EHR app triggers the logging transaction by calling the corresponding function of the smart contract.
- **Input**: timestamp, the Reason, the studyID and citizenPseudo, validation from individuals (e.g. S-EHR app signature), usage/access policy
- **Outcome**: A transaction registered in the ledger, with the input information.

**Smart Contract for Data Download Transaction**

- **Objective**: Researchers can check the transaction history and easily check what is shared and when.
- **Triggered by whom**: Researchers inside the same research center trigger the transaction to check the logging history.
- **Input**: studyID
- **Outcome**: The logged transaction with all the stored information.

The Blockchain API is a RESTful API that is automatically invoked by the aforementioned defined actors in order to execute the smart contract that depicts the data transactions. Services that require to perform actions on the blockchain will communicate with it using the endpoints that are provided by the API. To perform these tasks, the API relies on the Hyperledger Fabric Software Development Kit (SDK). Since the Hyperledger Fabric is a permissioned blockchain, it uses a CA to register participants and to allow them to generate a set of cryptographic keys. Two APIs are defined representing the two smart contracts. More details on the implementation aspects and the design will be provided in the final version of design of libraries for HR security and privacy services in **[**D3.11**]**.

**Operation dataUploadTransaction**

| | |
|---|---|
| **Name** | dataUploadTransaction |
| **Description** | This endpoint represents the smart contract for the data upload transaction and allows the S-EHR app to log in the blockchain the history of data sharing. This is a **POST** method to /dataUploadTransaction |
| **Arguments/Headers** | • HTTP Headers:<br>  ○ "Authorization": Citizen's Authorization token - JSON Web token<br>• Arguments:<br>  ○ Reason: String<br>  ○ studyID: String<br>  ○ citizenPseudo: String<br>  ○ validation: String |
| **Return Value** | HTTP Return Codes<br>• **200 Successful:** request was successfully processed.<br>• **400 Bad request:** request could not be processed.<br>• **401 Not Authorized**: authorization is required for the interaction that was attempted.<br>• **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |
| **Exceptions** | • Missing header: Authentication token |
| **Preconditions** | • N/A |

**Operation dataDownloadTransaction**

| | |
|---|---|
| **Name** | dataDownloadTransaction |
| **Description** | This endpoint represents the smart contract for the data and allows an authenticated user to read the blockchain transactions history. This is a **POST** method to /dataDownloadTransaction |
| **Arguments/Headers** | • HTTP Headers:<br>  ○ "Authorization": Citizen's Authorization token - JSON Web token<br>• Arguments:<br>  ○ studyID: String |
| **Return Value** | • transactions: String |

| | |
|---|---|
| | HTTP Return Codes<br>• **200 Successful:** request was successfully processed.<br>• **400 Bad request:** request could not be processed.<br>• **401 Not Authorized**: authorization is required for the interaction that was attempted.<br>• **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed. |
| **Exceptions** | • Missing header: Authentication token |
| **Preconditions** | • N/A |

# 4 ALIGNMENT WITH STANDARDS AND GUIDELINES

In all the security protocol specifications in the security deliverable **[D3.4]**, **[D3.6]** and the current D3.8, apart from the NIST recommendations in **[NIST 800-111]**, **[NISTDSS]** and ENISA recommendations in **[ENISA2021]**, several standards were also considered for secure communication of health data such as the **[ISO 13606-4]**, **[ISO 22857]** and **[ISO 22600-3]**. Below, a brief summary follows on how the aforementioned standards were utilized in the InteropEHRate project.

- The guide in **[NIST 800-111]** provides recommendations regarding storage encryption. All the recommendations were thoroughly analyzed to select the most appropriate crypto-primitive and approach. In the context of InteropEHRate AES 256 is used for symmetric encryption in all the defined protocols.
- The standard in **[NISTDSS]** provides information on the digital signatures schemes, from the generation to verification and validation. All the proposed digital signatures scheme and best practices are considered in the InteropEHRate security protocols.
- The recommendation in **[ENISA2021]**, provides information on the state-of-the-art pseudonymisation technique and a detailed analysis of the case each technique is necessary. Based on this recommendation we realised the second variant of the RDS protocol, where pseudonymisation can go beyond hiding real identities and data minimisation into supporting the unlinkability making high entropy pseudonyms necessary.
- The standard **[ISO 13606-4]** introduces a structure to classify access to healthcare data according to its sensitivity, the role of the person seeking access and purpose. This is also aligned to InteropEHRate consent management and authorization, where both the purpose and who is requesting access are clearly stated in the consent, while authorization is granted only after the consent is signed by the requestor to prove conen's authenticity.
- The standard **[ISO 22857]** addresses the protection requirements to facilitate cross-border transfer of personal healthcare data. In the context of InteropEHRate, both anonymisation techniques and consent related aspects are considered, as suggested from the guidelines.
- The standard **[ISO 22600-3]** principles and guidelines in a large systematic literature review of access control for electronic health record systems to protect patient's privacy. More precisely, defines a framework, privileges and obligations, but also consequences and penalties when the regulations are ignored. In addition, the standard includes information about patient's consent management. Several aspects regarding the privilege management, access control and consent management are considered, such as the policies for the ABAC mechanism in the S-EHR Cloud is inspired from this standard.

# 5 CONCLUSIONS AND NEXT STEPS

In this report, it is defined the second and final version of the specification of consent management and decentralized authorization mechanisms for HR exchange. A technical background with state-of-the-art mechanisms and standards was also provided. More specifically, this deliverable includes the detailed crypto models and consent management and authorization aspects of all the involved architecture components (e.g., S-EHR App, HCP Web App, S-EHR Cloud, Central Node and Reference Research Center), protocols (e.g., D2D, R2D Access, R2D Backup, R2D Emergency, RDS), and scenarios (e.g., Medical Visit, Emergency and Research). In general, consent management and authorization is done through digitally signed consent exchanges and access control mechanisms such as ABAC where necessary. In D2D, the citizen gives his/her signed content to authorize the healthcare organisation to download/upload his/her medical data. In R2D Access, access to retrieve their own data everywhere in Europe using their unique eIDAS identity is allowed only to authorized citizens. In R2D Backup and Emergency protocols, three consents have been specified. One mandatory, for authorizing S-EHR Cloud to store citizen's encrypted data and two optional for authorizing healthcare professionals to access citizen's data and for authorizing other healthcare organizations to access his/her DICOM images. In addition, an ABAC authorization mechanism is used for authorization of healthcare professionals in the Emergency protocol. Last but not least in RDS protocol two consents are used for participation of a citizen to a research study and for authorizing a research center to access citizen's medical data. Also, the specification of a blockchain-based scheme, as an optional service, for the logging transactions of research data sharing is provided. This final version of the deliverable acts as the detailed specification of consent management and authorization defined in the context of InteropEHRate.

# REFERENCES


- **[XACML]** OASIS, XACML Oasis Specification. Website: https://www.oasisopen.org/committees/tc_home.php?wg_abbrev=xacml
- **[NGAC]** NGAC Functional Architecture. Website: http://www.incits.org/scopes/INCITS499.htm
- **[APPC]** Integrating the Healthcare Enterprise, Advanced Patient Privacy Consents (APPC), Rev. 1.2 – Trial Implementation, 2018. Website: https://www.ihe.net/uploadedFiles/Documents/ITI/IHE_ITI_Suppl_APPC.pdf
- **[BPPC]** eHealth DSI Patient Summary and ePrescription, BPPC Profile, 2017.
- **[Bocek2018]** Bocek T., Stiller B. (2018) Smart Contracts – Blockchains in the Wings. In: Linnhoff-Popien C., Schneider R., Zaddach M. (eds) Digital Marketplaces Unleashed. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-49275-8_19
- **[Wüst2018]** K. Wüst and A. Gervais, "Do you Need a Blockchain?," *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, 2018, pp. 45-54, doi: 10.1109/CVCBT.2018.00011.
- **[CM1019]** Wikipedia, Consent management, 2019. Website: https://en.wikipedia.org/wiki/Consent_management
- **[FRASER1997]** Fraser, B, *Site Security Handbook*, IETF, 1997.
- **[FERRAIOLO2016]** David Ferraiolo, Ramaswamy Chandramouli, Rick Kuhn, and Vincent Hu. 2016. Extensible Access Control Markup Language (XACML) and Next Generation Access Control (NGAC). In Proceedings of the *2016 ACM International Workshop on Attribute Based Access Control (ABAC '16)*. ACM, New York, NY, USA, 13-24,2016, DOI: https://doi.org/10.1145/2875491.2875496
- **[Tandon2020]** Tandon, A.; Dhir, A.; Islam, A.K.M.N.; Mäntymäki, M. Blockchain in Healthcare: A Systematic Literature Review, Synthesizing Framework and Future Research Agenda. Computers in Industry 2020, 122, 103290, doi:https://doi.org/10.1016/j.compind.2020.103290.
- **[Meinert2019]** Meinert, E., Alturkistani, A., Foley, K.A., Osama, T., Car, J., Majeed, A., Van Velthoven, M., Wells, G., Brindley, D., 2019. Blockchain implementation in health care: protocol for a systematic review. JMIR Res. Protoc. 8 (2), e10994.
- **[Kuo2019]** Kuo, T.T., Gabriel, R.A., Ohno-Machado, L., 2019. Fair compute loads enabled by blockchain: sharing models by alternating client and server roles. J. Am. Med. Inform. Assoc. 26 (5), 392–403.
- **[Angraal2017]** Angraal, S., Krumholz, H.M., Schulz, W.L., 2017. Blockchain technology: applications in health care. Circ. Cardiovasc. Qual. Outcomes 10 (9), e003800.
- **[Mettler2016]** Mettler, M., 2016. Blockchain technology in healthcare: the revolution starts here. In: In 2016 IEEE 18th International Conference on E-Health Networking, Applications and Services (Healthcom), September, pp. 1–3.
- **[Hölbl2018]** Hölbl, M., Kompara, M., Kamisali ˇ c, ´ A., Nemec Zlatolas, L., 2018. A systematic review of the use of blockchain in healthcare. Symmetry 10 (10), 470.
- **[Ito2018]** Ito, K., Tago, K., Jin, Q., 2018. I-blockchain: a blockchain-empowered individual centric framework for privacy-preserved use of personal health data. In: In 2018 9th International Conference on Information Technology in Medicine and Education, (ITME, October), pp. 829–833.
- **[Alla2018]** Alla, S., Soltanisehat, L., Tatar, U., Keskin, O., 2018. Blockchain technology in electronic healthcare systems. IISE Annual Conf. Expo 2018 (1), 754–759.
- **[Swan2015]** Swan, M. Blockchain: Blueprint for a New Economy; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2015.

- **[Szabo1997]** Szabo N. Smart contracts: Formalizing and securing relationships on public networks. First Monday 1997;2(9). doi: 10.5210/fm.v2i9.548
- **[Biplov2020]** Biplov K.C., Blockchain in Healthcare, MSc thesis, 2020 https://kola.opus.hbz-nrw.de/opus45-kola/frontdoor/deliver/index/docId/2038/file/MasterThesis_Biplov_KC_216203865.pdf (Accessed 24 August 2021)
- **[Guardtime2019]** Guardtime, "Guardtime and estonian biobank announce a partnership to deploy guardtime helium" 2019 (Accessed 24 August 2021) https://guardtime.com/blog/guardtime-and-estonian-biobank-announce-apartnership-to-deploy-guardtime-helium
- **[Guardtime2019b]** Guardtime, "World's first blockchain-supported personal care record platform launched by guardtime and partners to up to 30 million nhs patients in the uk" 2019 (Accessed 24 August 2021) https://guardtime.com/blog/world-s-first-blockchain-supported-personal-care-record-platform-launched-by-guardtime-and-partners
- **[Dickson2018]** Dickson, B., 'Philips will challenge tech giants to bring blockchain to healthcare"., 2018 (Accessed 24 August 2021) https://thenextweb.com/blockchain/2018/10/17/philips-solve-healthcaredata-breaches-with-blockchain/
- **[Azaria2016]** Azaria, A., Ekblaw, A., Vieira, T. and Lippman, A., "Medrec: Using blockchain for medical data access and permission management", 2016 2nd International Conference on Open and Big Data (OBD) pp. 25–30.
- **[Allison2017]** Allison, I., "Gem shows off first blockchain application for health claims"., 2017 (Accessed 24 August 2021) https://www.ibtimes.co.uk/gem-shows-off-first-blockchain-applicationhealth-claims-1622574
- **[Prisco2016]** Prisco, G., "The blockchain for healthcare: Gem launches gem health network with philips blockchain lab", 2016 (Accessed 24 August 2021) https://bitcoinmagazine.com/articles/the-blockchain-for-heathcare-gemlaunches-gem-health-network-with-philips-blockchain-lab-1461674938
- **[Engelhardt2017]** Engelhardt M.A. Hitching Healthcare to the Chain: An Introduction to Blockchain Technology in the Healthcare Sector. Technol. Innov. Manag. Rev. 2017;7:22–34. doi: 10.22215/timreview/1111.
- **[Oyinloye2012]** Oyinloye, D.P.; Teh, J.S.; Jamil, N.; Alawida, M. Blockchain Consensus: An Overview of Alternative Protocols. Symmetry 2021, 13, 1363. https://doi.org/10.3390/sym13081363
- **[Nakamoto2008]** Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. https://doi.org/10.1007/s10838-008-9062-0.
- **[Clark2018]** Clark B., Burstall R. Blockchain, IP and the pharma industry—how distributed ledger technologies can help secure the pharma supply chain. J. Intell. Property Law Practice. 2018;13(7):531–533.
- **[Saad2020]** M. Saad et al., "Exploring the Attack Surface of Blockchain: A Comprehensive Survey," in IEEE Communications Surveys & Tutorials, vol. 22, no. 3, pp. 1977-2008, third quarter 2020, doi: 10.1109/COMST.2020.2975999.
- **[WADO-RS]** Web Access to DICOM Objects by RESTful Services, http://dicom.nema.org/medical/dicom/current/output/html/part18.html#sect_10.4 https://www.dicomstandard.org/dicomweb/retrieve-wado-rs-and-wado-uri
- **[Li2019]** Li G., Li L., Choi T.M., Sethi S.P. Green supply chain management in Chinese firms: Innovative measures and the moderating role of quick response technology. J. Oper. Manage. 2019 doi: 10.1002/joom.1061.
- **[Sheel2019]** Sheel A., Nath V. Effect of blockchain technology adoption on supply chain adaptability, agility, alignment and performance. Manage. Res. Rev. 2019;42(12):1353–1374.
- **[Betti2019]** Betti Q., Khoury R., Hallé S., Montreuil B. Improving hyperconnected logistics with blockchains and smart contracts. IT Prof. 2019;21(4):25–32.
- **[Forbes2019]** Forbes: Blockchain 50 (2019) Article URL, last (Accessed 31 June 2021)

- **[Androulaki2018]** Androulaki E., Barger A., Bortnikov V., Cachin C., Christidis K., De Caro A., et al. Hyperledger fabric: a distributed operating system for permissioned blockchains 13th EuroSys Conference, ACM, 2018
- **[Ethereum2020]** A next-generation smart contract and decentralized application platform, Ethereum Whitepaper, https://translatewhitepaper.com/wp-content/uploads/2021/04/EthereumOrijinal-ETH-English.pdf, last (Accessed 30 June 2021).
- **[Quorum2021]** Quorum whitepaper, last (Accessed 30 June 2021).
- **[Corda2021]** The corda platform: an introduction (2020) https://www.r3.com/wp-content/uploads/2019/06/corda-platform-whitepaper.pdf, last (Accessed 30 June 2021).
- **[D3.4]** InteropEHRate consortium. *D3.4 : Specification of remote and D2D IDM mechanisms for HRs Interoperability - V2*, 2021. www.interopehrate.eu/resources
- **[D3.6]** InteropEHRate consortium. *D3.6 : Specification of data encryption mechanisms for mobile and web applications - V2*, 2021. www.interopehrate.eu/resources
- **[D2.6]** InteropEHRate Consortium, *D2.6 - InteropEHRate Architecture - V3*, 2021. www.interopehrate.eu/resources
- **[D4.3]** InteropEHRate Consortium, *D3.4 - Specification of remote and D2D protocol and APIs for HR exchange V3*, 2021. www.interopehrate.eu/resources
- **[D4.9]** InteropEHRate Consortium, *D4.9 - Specification of protocol and APIs for research health data sharing - V2*, 2021. www.interopehrate.eu/resources
- **[D6.7]** InteropEHRate Consortium, *D6.7 - Design of a service for cloud storage of S-EHR content (S-EHR cloud)*, 2021. www.interopehrate.eu/resources
- **[D3.11]** InteropEHRate Consortium, D3.11 - -Design of libraries for HR security and privacy services - V3, 2021. www.interopehrate.eu/resources
- **[ANDROID2019]** Google, Android keystore system, Website: https://developer.android.com/training/articles/keystore
- **[APPLE2019]** Apple, Keychain Services, Website: https://developer.apple.com/documentation/security/keychain_services
- **[nist800-111]** NIST 800-111, Guide to Storage Encryption Technologies for End User Devices, Recommendations of the National Institute of Standards and Technology, 2007, https://www.hhs.gov/sites/default/files/nist800111.pdf
- **[NISTDSS]** FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, "Digital Signature Standard (DSS)", 2013, Website: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf
- **[ISO 13606-4]** ISO 13606-4, Health informatics — Electronic health record communication — Part 4: Security, 2019.
- **[ISO 22857]** ISO 22857, Health informatics — Guidelines on data protection to facilitate trans-border flows of personal health data. 2013.
- **[ISO 22600-3]** ISO 22600-3, Health informatics — Privilege management and access control — Part 3: Implementations, 2014.
- **[ENISA2021]** ENISA. "Pseudonymisation for Personal Data Protection". 2021.

# APPENDIX A

This section summarises all the notions used in the design of the cryptographic libraries and the JSON schemas for D2D requests.

| Symbol | Description |
|---|---|
| $G$ | Multiplicative group |
| $g$ | Generator |
| $Z_p^*$ | Group |
| $r$ | Random value |
| $q, p$ | Large primes |
| $\sigma$ | Cryptographic signature |
| $Pr$ | Private key |
| $C$ | Certificate |
| $Con$ | Concent |
| $Ver$ | Verify |
| $N$ | Nonce |
| $Z$ | Symmetric key |
| $Enc$ | Encryption |
| $Dec$ | Decryption |
| $SAMLRe$ | SAML Response |
| $Ext$ | Attribute extraction |
| $Auth$ | Authentication/Authorization |
| $m$ | Health data |

| | |
|---|---|
| $QR$ | QR code |
| $tstamp$ | Timestamp |
| $T_{ID}$ | Transient identifier - anonymous assertion |
| $L_{ID}$ | Long-term identifier - real id |
| $pid$ | Pseudo-id |
| $PGen$ | Pseudonym generation based on group signatures |
| $pseu$ | Pseudonym |
| $An$ | Anonymized the data with anonymous signing |
| $Map$ | Pseudonym mapping |

*Table 3 - Notation used*

## JSON-schema for the D2D Security Message

The JSON-schema for the D2D requests is specified below:

```
{
   "$id": "http://example.com/example.json",
   "$schema": "http://json-schema.org/draft-07/schema",
   "description": "The root schema of a D2DSecurityMessage",
   "required": [
      "header",
      "operation",
      "body"
   ],
   "type": "object",
   "properties": {
      "header": {
         "$id": "#/properties/header",
         "type": "object",
         "title": "The header schema",
         "description": "An explanation about the purpose of this instance.",
         "default": {},
         "examples": [
            {
               "timeStamp": "2021-07-26T14:13:13.553Z",
               "agent": "JRE 1.8.0_261 - Windows 10 10.0",
               "protocol": "D2D",
               "version": "1"
            }
         ],
         "required": [
            "timeStamp",
            "agent",
```

```
            "protocol",
            "version"
        ],
        "properties": {
            "timeStamp": {
                "$id": "#/properties/header/properties/timeStamp",
                "examples": [
                    "2021-07-26T14:13:13.553Z"
                ],
                "type": "string"
            },
            "agent": {
                "$id": "#/properties/header/properties/agent",
                "description": "The agent that created the message",
                "examples": [
                    "JRE 1.8.0_261 - Windows 10 10.0"
                ],
                "type": "string"
            },
            "protocol": {
                "$id": "#/properties/header/properties/protocol",
                "default": "D2D",
                "description": "The name of the used protocol.",
                "enum": [
                    "D2D"
                ],
                "type": "string"
            },
            "version": {
                "$id": "#/properties/header/properties/version",
                "default": "1",
                "description": "version of the protocol used",
                "type": "string"
            }
        },
        "additionalProperties": true
    },
    "operation": {
        "$id": "#/properties/operation",
        "description": "The name of the operation under execution of the D2D security protocol",
        "examples": [
            "HELLO_SEHR"
        ],
        "enum": [
            "HELLO_SEHR",
            "HELLO_HCP",
            "SEHR_PUBLIC_KEY",
            "HCP_PUBLIC_KEY",
            "UNSIGNED_CONSENT",
            "SIGNED_CONSENT"
        ],
        "type": "string"
    },
    "body": {
        "$id": "#/properties/body",
        "description": "The body of the message contains the exchanged data",
        "type": "string"
    }
```

```
    },
    "additionalProperties": true
}
```

## JSON sample for HCOConsent message

```
{
        "header": {
                "timeStamp": "2021-07-26T14:13:13.553Z",
                "agent": "JRE 1.8.0_261 - Windows 10 10.0",
                "protocol": "D2D",
                "version": "1"
        },
        "operation": "UNSIGNED_CONSENT",
        "body": "XTYRE8768LO8fwrqwm4l523k5203434279824jkhg2GTUYEbjgfg3232ljo9\u003d..."
}
```

## JSON sample for citizenSignedConsent message

```
{
        "header": {
                "timeStamp": "2021-07-26T14:13:14.553Z",
                "agent": "JRE 1.8.0_261 - Windows 10 10.0",
                "protocol": "D2D",
                "version": "1"
        },
        "operation": "SIGNED_CONSENT",
        "body": "MIHfMIGXBgkqhkiG9w0BAwEwgYkCQQD8poLOjhLKuibvzPcRDlJtsHiwXt3d..."
}
```