D3.4

Specification of remote and D2D IDM mechanisms for HRs Interoperability - V2

ABSTRACT

This deliverable provides the second and final version of the specification of remote and D2D Identity Management (IDM) including authentication mechanisms in InteropEHRate. This document also provides a detailed technical background for IDM and authentication mechanisms, which is a necessary step to move forward. The deliverable includes the IDM and authentication aspects of all the involved architecture components (e.g., S-EHR App, HCP Web App, S-EHR Cloud, 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 HRs interoperability.

Delivery Date	August 10th, 2021
Work Package	WP3
Task	T3.2
Dissemination Level	Public
Type of Deliverable	Report
Lead partner	UBIT

InteropEHRate project is co-funded by the European Union (EU) Horizon 2020 program under Grant number 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, Stella Dimopoulou	BYTE
	Alessio Graziani	ENG
Reviewers	Simone Bocca	UNITN
	Vincent Keunen	A7

LOGTABLE

Version	Date	Change	Author	Partner
0.1	12-05-21	First draft of ToC	Sofianna Menesidou	UBIT
0.2	18-05-21	Section 3 restructuring	Sofianna Menesidou	UBIT
0.3	27-05-21	Section 3	Sofianna Menesidou	UBIT
0.4	28-05-21	Section 3	Sofianna Menesidou	UBIT
0.5	31-05-21	Section 3	Sofianna Menesidou	UBIT
0.6	03-06-21	Section 3	Sofianna Menesidou	UBIT
0.7	07-06-21	Early review	Simone Bocca	UNITN
0.8	08-06-21	R2D Backup, R2D Emergency APIs	Chrysostomos Symboulidis	BYTE
0.9	09-06-21	Section 3	Sofianna Menesidou	UBIT
1.0	23-06-21	Section 3	Sofianna Menesidou	UBIT
1.1	24-06-21	Section 3	Sofianna Menesidou	UBIT
1.2	28-06-21	Section 3	Sofianna Menesidou	UBIT
1.3	29-06-21	Section 3, Conclusions	Sofianna Menesidou	UBIT

1.4	06-07-21	Quality check	Argyro Mavrogiorgou	UPRC
1.5	04-08-21	Section 3 updates based on comments	Sofianna Menesidou	UBIT
VFInal	09-08-21	Final tech revision and version for submission	Francesco Torelli Laura Pucci	ENG

ACRONYMS

Acronym	Term and definition
2FA	Two-Factor Authentication
ABAC	Attribute-based Access Control
B2C	Businesses and Consumers
B2E	Businesses and Employees
BLE	Bluetooth Low Energy
CEF	Connecting Europe Facility
eHDSI	eHealth Digital Service Infrastructure
eID	Electronic identification
ETL	European Trust Lists
FHIR	Fast Healthcare Interoperability Resources
G2C	Governments and Citizens
HO	Health Organisation
HSM	Hardware Security Modules
IDM	Identity Management
LSP	Large Scale Pilots
MS	Middleware-Service
NFC	Near-Field Communication
NCP	National Contact Point
OIDC	OpenID Connect
OCSP	Online Certificate Status Protocol
PKI	Public Key Infrastructure
PI	Principal Investigator
PP	Pseudonym Provider
QC	Qualified Certificates

QSCD	Qualified Signature Creation Device
RRC	Reference Research Center
SAML	Security Assertion Markup Language
SP	Service Provider
TSP	Trusted Service Provider
U2F	Universal 2nd Factor Authentication
UAF	Universal Authentication Framework
USB	Universal Serial Bus
WebAuthN	Web Authentication API
WS-Federation	Web Services Federation
XACML	eXtensible Access Control Markup Language

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

Electronic identification (eID) within the eHealth domain is the cornerstone of patient safety. Improvements in electronic identification systems and processes have emerged as a fundamental concern for European states in their pursuit of better relationships and interactions between governments and citizens (G2C), businesses and consumers (B2C), and businesses and employees (B2E) **[KAI2009]**. Specific regulations are being established to ensure that electronic signatures provide a legal standing equivalent to that of handwritten signatures. eIDAS **[EE2017]** and NIST-DSS **[NISTDSS]** for the EU and USA respectively are examples of such regulations.

The eIDAS **[EE2017]** Regulation basically establishes a cross-border and cross-sector legal framework that covers electronic signatures, electronic documents, electronic time stamps, electronic seals, certificate services, and electronic registered delivery services in relation to eIDs and EU-based trust service providers. Trust forms the heart of the eIDAS Regulation, where eID is short for electronic ID. The regulations for electronic IDs have come into force from autumn of 2018, which is closely related to the cross-border acceptance of eIDs in public services **[NGUYEN2018]**. The European Commission is using eIDAS as a toolbox to ensure the trustworthiness of various online services that fall under the control of the Commission.

This report provides the final version of the specification of remote and D2D Identity Management (IDM) including authentication mechanisms in InteropEHRate considering the regulations and the state-of-the-art mechanism for interoperable eIDs considering aspects of all the involved architecture components (e.g., S-EHR App, HCP Web App, S-EHR Cloud, 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 HRs interoperability.

1.1 Scope of the document

The main goal of the present document is to describe the InteropEHRate specification of remote and D2D Identity Management (IDM) and authentication mechanisms for HRs interoperability. Moreover, the deliverable describes the research conducted regarding identity management and authentication mechanisms. In a nutshell, identity management and authentication is done in most scenarios through Certificates acquired from a CA, while in the context of R2D Access and in the second variant of RDS scenario an eIDAS-based solution to achieve cross-border interoperability is specified. IDM and authentication in R2D Backup and Emergency is achieved by a simple username/password login and attribute-based access control (ABAC) respectively to the cloud provider.

1.2 Intended audience

The 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 protocols support identity management and authentication mechanisms for health records interoperability.

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 regarding identity management and authentication, starting with a general overview and then focusing on other related European research initiatives.

- **Section 3** introduces the overall identity management and authentication mechanisms in terms of InteropEHRate considering all the different protocols and scenarios. This section includes the security models for all the security protocols to highlight the used crypto-primitives.

- **Section 4** concludes the deliverable and highlights the most important aspects of the solutions used.

- **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 identity management and authentication mechanisms per protocol is included and described in the deliverable.
- 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.
- Updated sequence diagrams including the R2D Access authentication based on eIDAS is provided and described thoroughly in the following steps.
- The section “relations to other deliverables” for similarity with other deliverables has been removed.
- 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 and the JSON schemas for D2D requests are included in the deliverable.

2 TECHNICAL BACKGROUND

This chapter presents an understanding of the state-of-the-art concerning identity management (IDM) including authentication mechanisms for interoperability in the healthcare domain, based on a review of the current literature. Background and more details regarding authorization mechanisms (e.g. ABAC or attribute-based access control) will be provided in the **[D3.8]**. Even though authorization is closely tied to authentication and IDM, it will be omitted from this section since **[D3.8]** is a more appropriate deliverable for these aspects.

2.1 State of the art in IDM

A Digital Identity is the information used to represent an entity in an ICT system **[IT2011]**. An entity may be a person, an organization, a device, an application, etc. Electronic Identification provides the proper authentication strength for patients when seeking health care in a cooperating EU member state, as well as safeguarding their fundamental access rights.

Identity management (IDM) is the mechanism or objects used by entities to manage the claims about their digital identities. Working on identity management in the health area is not reduced to unique identification of citizens/patients, but also of healthcare professionals and health institutions. Personal health data are handled as explicit sensitive data and the definition and management of rights is essential for reaching a status which conforms to the legal systems in the member states **[EU2009]**.

Authentication is a security mechanism that allows systems to validate the user as a registered user by providing information to prove the user is who he/she claims to be. There are several authentication mechanisms in the literature based on biometrics, usernames and passwords, certificates, tokens, etc. The most common mechanism is the combination of username and password. Some alternatives are HTTP-based authentication by using HTTP headers and other more modern approaches include two-factor authentication and password-less mechanisms. We will analyse the most important of them in the sub-section below.

The architecture of identity management (IDM) systems can be divided in two distinct categories according to **[SCUDDER2010]**:

- **Network Based IDM** - In this category, the attributes are stored at the identity provider and users authorize a request by the relying party to access the attributes. The relying party can then access the data from the identity provider.
- **Claim Based IDM** - In this category, the attributes are stored at the user. A relying party can request the user to show the possession of these claims. The user can then use these claims to directly interact with the relying party, without any additional interaction with the issuer.

The most known identity management models are briefly detailed below **[CARRETERO2018]**:

- **Isolated Model** - In this model the Service Provider (SP) and Identity Provider (IdP) are combined in a single server. However, this model is a very simple approach and may cause many problems **[JOSANG2005]**.
- **Centralized Model** - This model consists of centralizing the identity storage while separating the services. Multiple SPs have to authenticate their users against a central IdP. The most extended implementation is the single sign-on (SSO) authentication method, which enables the user to

access several SPs with a single identification instance. This model reduces the usability problems derived from the Isolated Model, but has a clear reliability problem, as it depends on a single point of failure **[CARRETERO2018]**.

- **Federated Model** - In this model, the parties involved in the identity management system establish an agreement on which entities are part of the system, how entities are going to be referred to, and the configuration parameters of the participating system parties. A Service Provider in one domain can grant authorized access to a resource it manages based on the exchange of identity, attribute, authentication and authorization assertions with an Identity Provider in another domain.

2.1.1 Identity Federations

There are several Identity Federations in the education, government and research sectors as well as the general public. According to **[EMTG2017]** the most known hybrid Identity Federations in the literature are the STORK and eIDAS. Identity federations are based on the establishment of trust agreements between organizations. Thus, any user in the federation will be able to access resources and services of any federated organization based on a unique digital identity, which is common to the whole federation. This federated identity has two benefits: a) simplifies the credential control by the user and b) the user management by service providers **[EMTG2017]**. Within identity federations, there are different types of entities that interact with end users, the Service Providers and Identity Providers. In general, Identity Providers include the Authentication Provider and the Attribute Provider **[EMTG2017]**.

eIDAS Regulation **[EE2017]** on electronic identification and trust services for electronic transactions in the European internal market is based on the work done along the STORK and STORK 2.0 projects and it has been designed as an evolution of both of them. The eIDAS Regulation was published in 2014 as a regulatory environment that guarantees people and services the use of their national eIDs to use public services in all European countries with the same legal reliability as traditional paper based procedures. eIDAS promotes and facilitates the use of cross-border electronic identification and trust services, and guarantees transparency and accountability. The objective is to extend and popularize the use of eID among citizens of the European Union in their relations with institutions as well as in the private sector.

2.1.2 Electronic Identification (eID) and the eHealth domain

eIDAS defines citizens as persons and organisations that seek online services from any EU member state using their domestic eID with assured security, cost- and time-efficiencies, and usability **[KENNEDY2016]**. eIDAS eID consolidates the independent national eID schemes by streamlining their output through Nodes and Connectors. The proprietary national input is mapped and conditioned through the eIDAS Node in Country-A to an interoperable transport form, the eIDAS SAML Assertion. Such assertions can be requested during an authentication request by a Service Provider (SP) through an eIDAS Connector in Country-B **[ESENS2017]**.

A typical eID ecosystem comprises of **[KENNEDY2016]**:

- **citizens,**
- **member states**,
- **node operators** or connection points,
- **attribute and identity providers** that provide information related to electronic identities and that verify user identities and

- **service providers** that offer online services whose access is authenticated through eID.

Electronic identification within the eHealth domain is motivated by two primary goals a) **patient safety** and b) **protection against illegitimate disclosure of medical data**, while it is identified as a difficult task. The work in **[KATEHAKIS2017]** already summarizes the main difficulties for eID, such as technical issues that hinder an efficient eID due to the need of hardware devices and middleware, the inability to reach foreign security services, or the inability to deploy non-certified software onto highly regulated medical systems are some of the identified issues and the maintenance of the legitimate-use of a national eID means across borders. The use of cross-border authentication through the eIDAS Network provides a reliable, responsible and convenient manner for online-services to identify their users **[eIDAS2018]**.

Important groundwork on the interoperability of EHRs was carried out in the framework of the project epSOS, and with the support of the EXPAND project paved the way to roll out of the eHealth Digital Service Infrastructure (eHDSI). Currently, the most important initiative for interoperability at European level is eHealth Digital Service Infrastructure (eHDSI or eHealth DSI), that offers initial services for cross-border health data exchange under the Connecting Europe Facility (CEF). These services are limited to the exchange of four kinds of documents namely ePrescription and eDispensation, Patient Summary, European Reference Networks, and Patient Registries.

2.1.3 eIDAS Infrastructure

The eIDAS interoperability framework comprises two different authentication models **[BERBECARU2019]**.

- In the **proxy model**, each country adhering to this model has to run a single national bridge called eIDAS node. This element is actually composed of two logical subcomponents:
  - an eIDAS-Proxy-Service (eIDAS Proxy), which is in charge of communicating with the National eID scheme to which the citizen will be authenticated; and
  - one eIDAS-Connector (Connector), which is in charge of communicating with the national SPs.
- The **middleware model** (adopted by Germany) does not exploit a national bridge:
  - the eIDAS Connector (in the other countries) communicates with a country-specific Middleware-Service (MW) to allow SPs to provide eIDAS-enabled services to German citizens. Citizen authentication is delegated from an SP to its national Connector, which acts as a gateway and subsequently forwards the authentication request to the eIDAS Proxy of the country selected by the citizen (or to the MW). The authentication request is further handled by the eIDAS Proxy according to Member State (MS)-specific approach.

Most countries follow the traditional approach, in which a new authentication request is constructed by the eIDAS Proxy and is sent (through the user's browser) to the national IdP (part of the National eID scheme). At the IdP, the citizen is asked to authenticate with a national eID. If this operation completes successfully, an authentication response containing also the eIDAS attributes that have been requested are returned through the eIDAS infrastructure back to the requesting SP.

Each eIDAS node has a Specific part used to communicate with the national SPs and IdPs and a Generic part used to communicate with the other eIDAS nodes via the eIDAS communication protocol, which is based on SAML 2.0 WebSSO Profile to transfer authentication data and eIDAS attributes between the eIDAS nodes.

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. The attributes are either mandatory or optional.

2.1.4 Identity and Authentication Standards

In general, there are two types of standards: "build it and they will come" standards (aka de facto standards), and "let's work together so we don't all do something different" standards (aka de jure standards). Below we cover the most known and used identity and authentication standards. A timeline of the standards is also presented in Figure 1 below, in order to have a more holistic view.

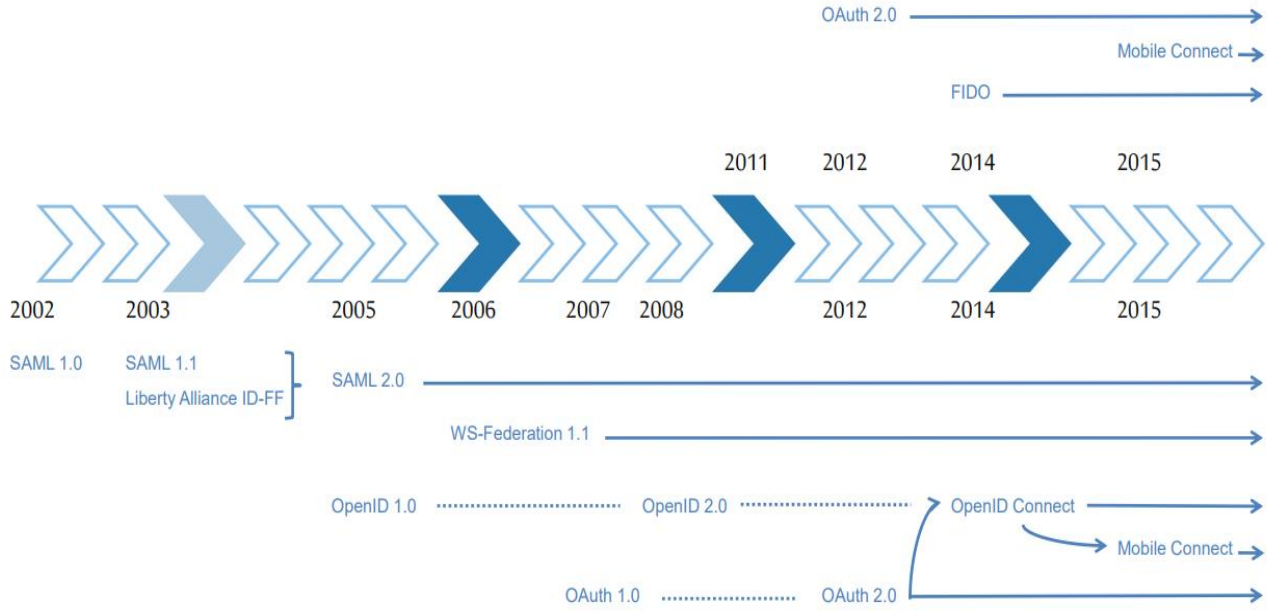

Figure 1 - Standards Timeline

2.1.4.1 SAML

Security Assertion Mark-up Language (SAML) is an XML-based open standard for exchanging authentication and authorization data between parties, in particular, between an identity provider and a service provider. This security information is expressed in the form of portable SAML assertions that applications working across security domain boundaries can trust. The OASIS SAML standard defines precise syntax and rules for requesting, creating, communicating, and using these SAML assertions. The four main components of the standard are the Assertions, Protocols, Bindings and Profiles.

SAML is one of the most important web-based federated identity standards. It's the most widely supported standard by SaaS providers who want to accept credentials from large enterprise customers. Like most other federated identity standards, it is based on redirecting a person's browser to a website maintained by their home organization. Assuming the website is trusted, the home organization then returns information about the person to the original website **[SAML2005]**. In this standard, the Identity providers pass identity information to service providers through digitally signed XML documents.

The SAML specification defines three roles: a) the principal (e.g. user), the identity provider (IdP), and the service provider (SP). In the primary use case addressed by SAML, the principal requests a service from the service provider. The service provider requests and obtains an authentication assertion from the identity provider. On the basis of this assertion, the service provider can make an access control decision, that is, it can decide whether to perform the service for the connected principal. In SAML, one identity provider may provide SAML assertions to many service providers. Similarly, one SP may rely on and trust assertions from many independent IdPs. The SAML Web Browser SSO profile was specified and standardized to promote interoperability.

A "SAML assertion" is a statement written in XML and issued by an "identity provider" about a "subject" (person) for a "relying party" (the recipient of the assertion) who is normally a "service provider" (website). Identity provider is abbreviated simply as "IDP" and service provider as "SP". SAML is a mature standard, and it's been successfully deployed to solve many business challenges. SAML uses public key cryptography to sign or encrypt messages and documents. The use of such keys enables the parties to protect and verify the integrity of information **[SAML2]**. SAML anonymous assertion will be utilised for user authentication in the Pseudonym Provider in the second variant of RDS scenario, as well as SAML assertion will be acquired in the RDS Access scenario from the eIDAS Node. More details will be provided in Section 3.

Assertions contain the information that a web application needs from the Identity Provider about the person accessing the site. A typical example of a SAML assertion presented below.

```
<saml:Assertion
   xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   ID="b07b804c-7c29-ea16-7300-4f3d6f7928ac"
   Version="2.0"
   IssueInstant="2004-12-05T09:22:05Z">
   <saml:Issuer>https://idp.example.org/SAML2</saml:Issuer>
   <ds:Signature
     xmlns:ds="http://www.w3.org/2000/09/xmldsig#">...</ds:Signature>
   <saml:Subject>
     <saml:NameID
       Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient">
       3f7b3dcf-1674-4ecd-92c8-1544f346baf8
     </saml:NameID>
     <saml:SubjectConfirmation
       Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
       <saml:SubjectConfirmationData
         InResponseTo="aaf23196-1773-2113-474a-fe114412ab72"
         Recipient="https://sp.example.com/SAML2/SSO/POST"
         NotOnOrAfter="2004-12-05T09:27:05Z"/>
     </saml:SubjectConfirmation>
   </saml:Subject>
   <saml:Conditions
     NotBefore="2004-12-05T09:17:05Z"
     NotOnOrAfter="2004-12-05T09:27:05Z">
     <saml:AudienceRestriction>
       <saml:Audience>https://sp.example.com/SAML2</saml:Audience>
     </saml:AudienceRestriction>
   </saml:Conditions>
   <saml:AuthnStatement
```

```
    AuthnInstant="2004-12-05T09:22:00Z"
    SessionIndex="b07b804c-7c29-ea16-7300-4f3d6f7928ac">
    <saml:AuthnContext>
      <saml:AuthnContextClassRef>
        urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
      </saml:AuthnContextClassRef>
    </saml:AuthnContext>
  </saml:AuthnStatement>
  <saml:AttributeStatement>
    <saml:Attribute
      xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
      x500:Encoding="LDAP"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
      Name="urn:oid:1.3.6.1.4.1.5923.1.1.1.1"
      FriendlyName="eduPersonAffiliation">
      <saml:AttributeValue
        xsi:type="xs:string">member</saml:AttributeValue>
      <saml:AttributeValue
        xsi:type="xs:string">staff</saml:AttributeValue>
    </saml:Attribute>
  </saml:AttributeStatement>
</saml:Assertion>
```

SAML Architecture

The core SAML specification defines the structure and content of both a) assertions and b) protocol messages used to transfer this information [SAML2005].

- SAML assertions carry statements about a principal that an asserting party claims to be true. The valid structure and contents of an assertion are defined by the SAML assertion XML schema.
- SAML protocol messages are used to make the SAML-defined requests and return appropriate responses. The structure and contents of these messages are defined by the SAML-defined protocol XML schema.

SAML profiles are defined to satisfy a particular business use case, for example the Web Browser SSO profile. Profiles typically define constraints on the contents of SAML assertions, protocols, and bindings in order to solve the business use case in an interoperable fashion. There are also Attribute Profiles, which do not refer to any protocol messages and bindings, that define how to exchange attribute information using assertions in ways that align with a number of common usage environments. Figure 2 illustrates the relationship between these basic SAML concepts [SAML2005].

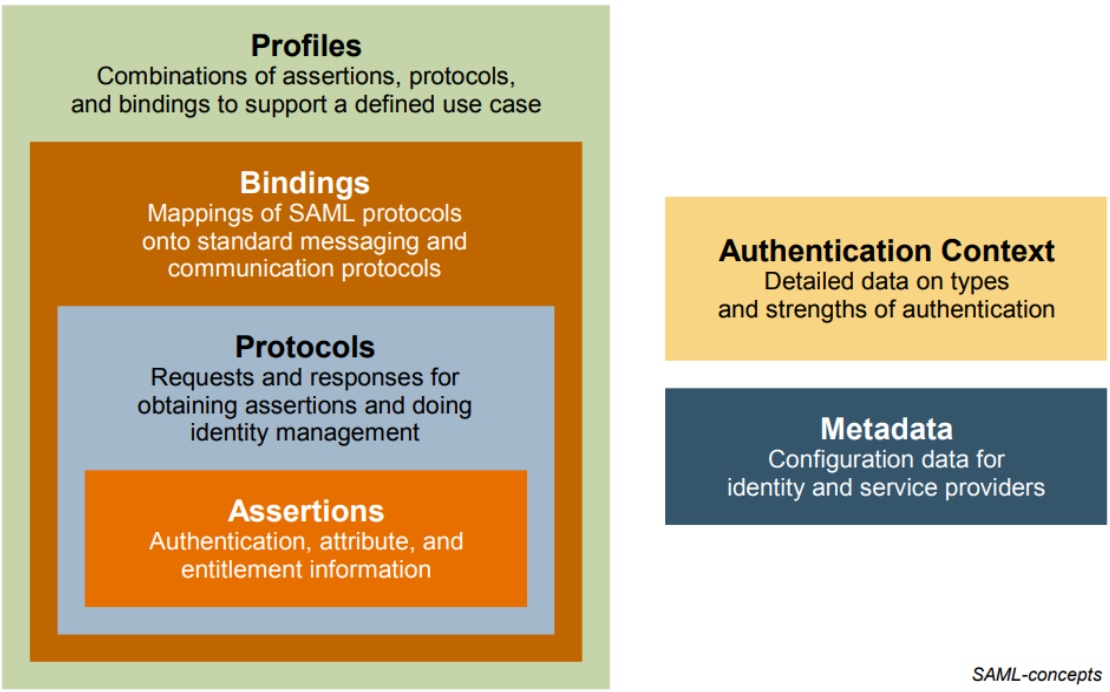

Figure 2 - Basic SAML Concepts

Two other SAML concepts are useful for building and deploying a SAML environment are a) Metadata and b) Authentication Context. Metadata defines a way to express and share configuration information between SAML parties, while a SAML authentication context is used in an assertion's authentication statement to carry information regarding the type and strength of authentication that a user employed when they authenticated at an identity provider.

The Liberty Alliance Project released frameworks for federation, identity assurance, an Identity Governance Framework, and Identity Web Services. Liberty endorses SAML2 as its identity federation solution and provides interoperability and conformance testing.

2.1.4.2 WS-Federation

WS-Federation (Web Services Federation) is an Identity Federation specification, developed by a group of companies: BEA Systems, BMC Software, CA Inc. (along with Layer 7 Technologies now a part of CA Inc.), IBM, Microsoft, Novell, HP Enterprise, and VeriSign. Part of the larger Web Services Security framework, WS-Federation defines mechanisms for allowing different security realms to broker information on identities, identity attributes and authentication **[WSFEDERATION]**.

WS-Security, WS-Trust, and WS-SecurityPolicy provide a basic model for federation between Identity Providers and Relying Parties. These specifications define mechanisms for codifying claims (assertions) about a requestor as security tokens which can be used to protect and authorize web services requests in accordance with policy. WS-Federation extends this foundation by describing how the claim transformation model inherent in security token exchanges can enable richer trust relationships and advanced federation of services. This enables high value scenarios where authorized access to resources managed in one realm can be provided to security principals whose identities and attributes are managed in other realms. WS-Federation includes mechanisms for brokering of identity, attribute discovery and retrieval, authentication

and authorization claims between federation partners, and protecting the privacy of these claims across organizational boundaries.

A federation is a collection of realms (security domains) that have established relationships for securely sharing resources. A Resource Provider in one realm can provide authorized access to a resource it manages based on claims about a principal (such as identity or other distinguishing attributes) that are asserted by an Identity Provider (or any Security Token Service) in another realm.

The value of establishing a federation is to facilitate the use of security principal attributes across trust boundaries to establish a federation context for that principal. A Relying Party can then use this context to grant/deny access to a resource. Establishing a federation context when Identity and Resource Providers operate in different realms requires agreement between these parties on what claims are required and frequently requires agreement on mechanisms for securely transporting those claims over unprotected networks. This provides the basis for interoperability. In general it is necessary for participants in a federation to communicate these requirements over a wide variety of trust and communication topologies. Supporting different topologies requires the exchange of metadata describing endpoint references where services may be obtained, plus the potential security policies and communication requirements that must be observed when accessing those endpoints. The exchange of this metadata can be further complicated because the participants in a single federation may have different policies and service providers may participate in multiple federations.

2.1.4.3 OAuth 2.0

OAuth was introduced to allow a user to grant access to private resources connected to their identity and is a standard for authorization and a set of defined process flows for "delegated authorization". OAuth 2.0 is a specification as to how to issue access tokens. It is defined in RFC 6749 (The OAuth 2.0 Authorization Framework) **[RFC6749]**. This is done without sharing private identity details or passwords between services. A long-lasting access token is specified by the protocol, which can be used by entities for continued access to user resources **[OAUTH2010]**.

OAuth is distinct from OpenID, as although it shares the common architecture of redirection for obtaining authorization, it only manages the access control of resources. OAuth and its updated standard OAuth 2.0 are both still in active use by many social networks and dependent applications. It uses JSON as the data format, and RESTful APIs to enable a person (or organization) to authorize access to resources. OAuth is a **delegated authorization protocol, not an authentication protocol**. OAuth is used in a wide variety of applications, including providing mechanisms for user authentication. This has led many developers and API providers to incorrectly conclude that OAuth is itself an authentication protocol and to mistakenly use it as such.

OAuth 2.0 can be extended by implementing custom grant types and/or token types. Some of the profiles built on top of the core framework are:

- SAML 2.0 Bearer Assertion Profile - used to exchange SAML assertions for access tokens
- User Managed Access Profile - enables the resource owner to define and manage multiple access policies for his protected resources in a single place
- Chain Grant Type Profile - enables a resource service to use the received access token

- Token Introspection Profile - allows clients to request metadata regarding a token
- Token Revocation Profile - used to revoke a token
- Dynamic Client Registration Profile - allows clients to register with an authorization server and retrieve their client ID and secret dynamically

OAuth Protocol Flow

The abstract OAuth 2.0 flow illustrated in the Figure below, which describes the interaction between the four roles and includes the following steps **[RCF6749]**:

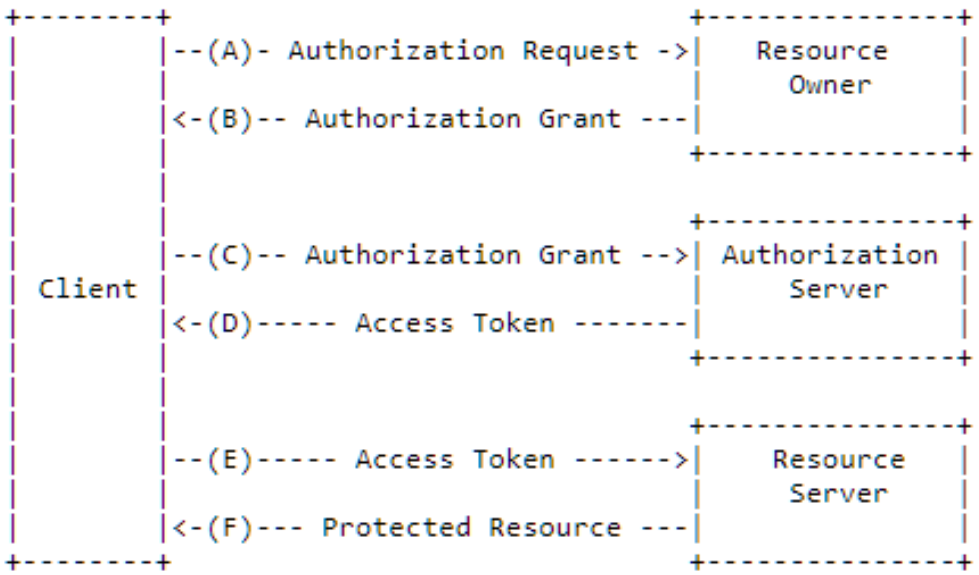

Figure 3 - Abstract Protocol Flow

A. The client requests authorization from the resource owner.
B. The client receives an authorization grant, which is a credential representing the resource owner's authorization, expressed using one of four grant types defined in this specification or using an extension grant type.
C. The client requests an access token by authenticating with the authorization server and presenting the authorization grant.
D. The authorization server authenticates the client and validates the authorization grant, and if valid, issues an access token.
E. The client requests the protected resource from the resource server and authenticates by presenting the access token.
F. The resource server validates the access token, and if valid, serves the request.

2.1.4.4 OpenID / OpenID Connect (OIDC)

OpenID is an open standard for authentication, promoted by the non-profit OpenID Foundation. There are over a billion OpenID-enabled accounts on the internet, and organizations such as Google, WordPress, Yahoo, and PayPal use OpenId to authenticate users. OpenID allows users to create an account with an identity provider that supports the standard, known as the OpenID Provider **[FITZPATRICK2005]**. A user must obtain an OpenID account through an OpenID identity provider (e.g. Google). The user will then use

that account to authenticate - sign into any website that accepts OpenID authentication, without managing multiple usernames and passwords. The OpenID standard provides a framework for the communication that must take place between the identity provider and the relying party.

In OpenID, authentication is delegated: server A wants to authenticate user U, but U's credentials (e.g. U's name and password) are sent to another server, B, that A trusts (at least, trusts for authenticating users). Indeed, server B makes sure that U is indeed U, and then tells A: "ok, that's the genuine U". Basically, OpenID is about verifying a person's identity. OpenID removes the requirement for remembering passwords across many sites, but still leaves the users trusting their OpenID identity provider with important data. This inherent centralisation, coupled with the fact that users are forced to rely on an abstract identity system, eventually caused OpenID to lose prominence on the web **[OPENID2011]**.

The latest version of OpenID is OpenID Connect, which combines OpenID authentication and OAuth2 authorization. OpenID Connect combines the features of OpenID 2.0, OpenID Attribute Exchange 1.0, and OAuth 2.0 in a single protocol. It allows an application to use authority a) to verify the end user's identity, b) to fetch the end user's profile info, and c) to gain limited access to the end user's stuff. Is an open standard for authentication and a set of defined process flows for “federated authentication”. OpenID Connect implements an authentication layer on top of the OAuth 2.0 protocol and employs REST/JSON for messaging. It is a “profile” of OAuth 2.0 specifically designed for attribute release and authentication. It allows clients of all types, including Web-based, mobile, and JavaScript clients, to verify the identity of the end-user based on the authentication performed by an Authorization Server, as well as to request and receive information about authenticated sessions. OpenID Connect is a suite of lightweight specifications that provide a framework for identity interactions via REST like APIs. OpenID Connect Clients use the scope values as defined in OAuth 2.0 to specify what access privileges are requested for Access Tokens. The scopes associated with Access Tokens determine what resources will be available when they are used to access OAuth 2.0 protected endpoints **[ALTICELABS2014]**. OpenID Connect has many parallels to SAML. The equivalent of the SAML assertion is an id_token , a signed JSON Web Token, or JWT that contains very similar information.

The OpenID Connect specification defines three roles:

- The **end user** or the entity that is looking to verify its identity
- The **relying party** (RP), which is the entity looking to verify the identity of the end user
- The **OpenID Connect provider** (OP), which is the entity that registers the OpenID URL and can verify the end user’s identity

OpenID Connect Authorization Code Flow

The diagram below depicts at a high level the Authorization Code Flow.

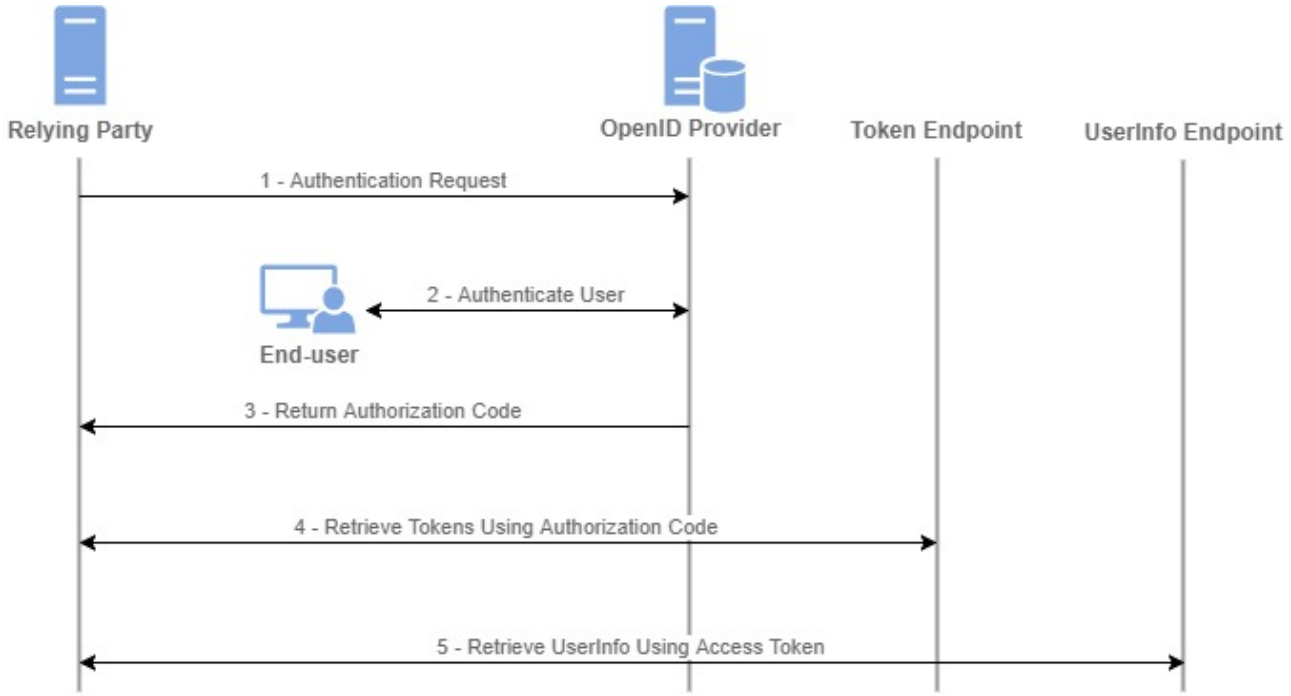

Figure 4 - High-level Authorization Code Flow

1. The Relying Party sends a request to the OpenID Provider to authenticate the End-User.
2. The OpenID Provider authenticates the end-user using one of the methods available to it and obtains authorization from the End-user to provide the requested scopes to the identified Relying Party.
3. Once the End-User has been authenticated and has authorized the request the OpenID Provider will return an authorization code to the Relying Party's server component.
4. The Relying Party's server component contacts the token endpoint and exchanges the authorization code for an id token identifying the end-user and optionally access and refresh tokens granting access to the userinfo endpoint.
5. Optionally the Relying Party may request the additional user information from the userInfo endpoint by presenting the access token obtained in the previous step.

2.1.4.5 FIDO Universal Authentication Framework (UAF)

The FIDO (Fast Identity Online) UAF strong authentication framework hosted by FIDO Alliance and enables online services and websites, whether on the open Internet or within enterprises, to transparently leverage native security features of end-user computing devices for strong user authentication and to reduce the problems associated with creating and remembering many online credentials **[UAF2017]**. More precisely, FIDO UAF supports a passwordless experience.

The user carries a device with a FIDO UAF stack installed. Users can then register their device to the online service by selecting a local authentication mechanism such as swiping a finger, looking at the camera, speaking into the mic, entering a PIN, etc. The FIDO UAF protocol allows the service to select which mechanisms are presented to the user. Once registered, the user simply repeats the local authentication action whenever they need to authenticate to the service. The user no longer needs to enter their

password when authenticating from that device. FIDO UAF also allows experiences that combine multiple authentication mechanisms such as fingerprint and PIN.

FIDO UAF High-Level Architecture

The FIDO UAF Architecture is designed to meet the FIDO goals and yield the desired ecosystem benefits **[UAF2017]**. It accomplishes this by filling in the status-quo's gaps using standardized protocols and APIs. The following Figure 5 summarizes the reference architecture and how its components relate to typical user devices and Relying Parties.

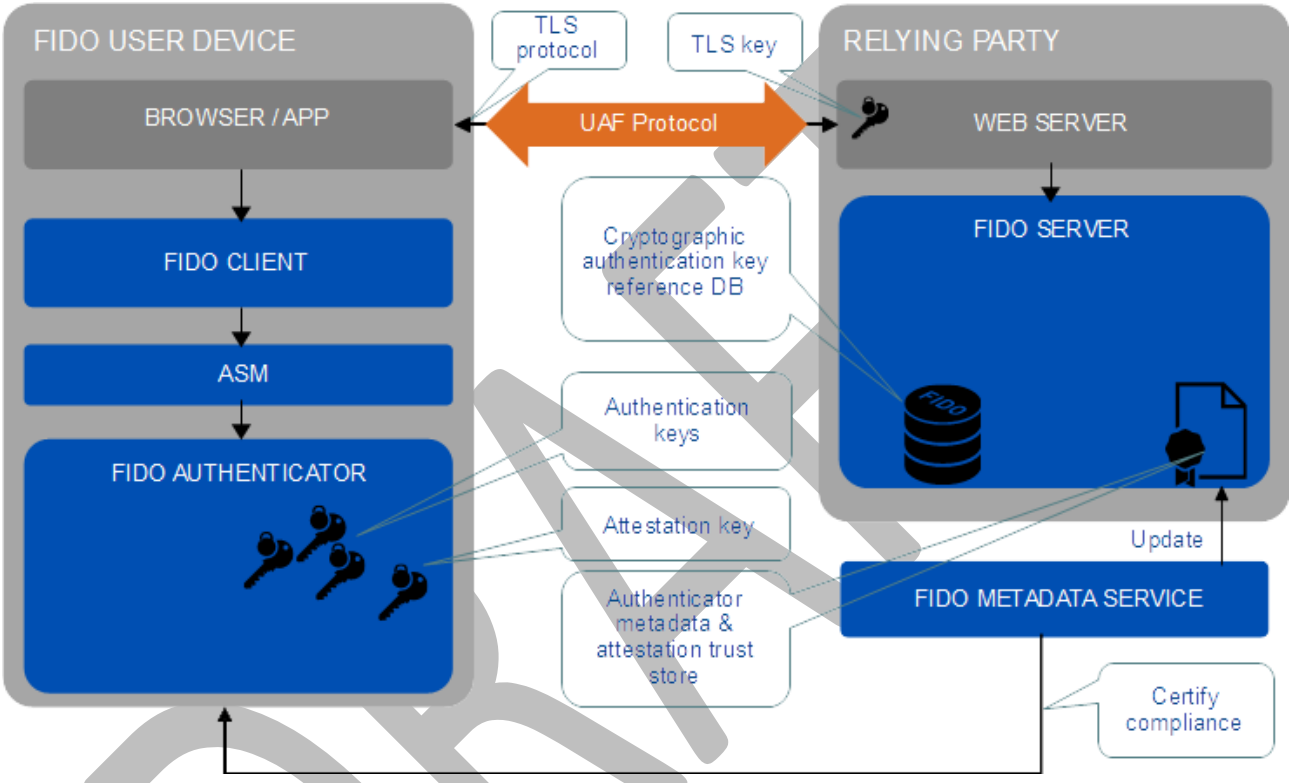

Figure 5 - FIDO UAF High-Level Architecture

A **FIDO UAF Client** implements the client side of the FIDO UAF protocols, and is responsible for a) interacting with specific FIDO UAF Authenticators using the FIDO UAF Authenticator Abstraction layer via the FIDO UAF Authenticator API and b) interacting with a user agent on the device (e.g. a mobile app, browser) using user agent-specific interfaces to communicate with the FIDO UAF Server.

A **FIDO UAF Server** implements the server side of the FIDO UAF protocols and is responsible for a) interacting with the Relying Party web server to communicate FIDO UAF protocol messages to a FIDO UAF Client via a device user agent, b) validating FIDO UAF authenticator attestations against the configured authenticator metadata to ensure only trusted authenticators are registered for use, c) manage the association of registered FIDO UAF Authenticators to user accounts at the Relying Party and d) evaluating user authentication and transaction confirmation responses to determine their validity.

A **FIDO UAF Authenticator** is a secure entity, connected to or housed within FIDO user devices, that can create key material associated with a Relying Party. The key can then be used to participate in FIDO UAF strong authentication protocols.

FIDO UAF Protocol Message Flows

The FIDO UAF Protocols carry FIDO UAF messages between user devices and Relying Parties. There are protocol messages addressing a) Authenticator Registration, b) User Authentication, c) Secure Transaction Confirmation and d) Authenticator Deregistration. Figure 6, below presents the UAF Authentication Sequence Diagram.

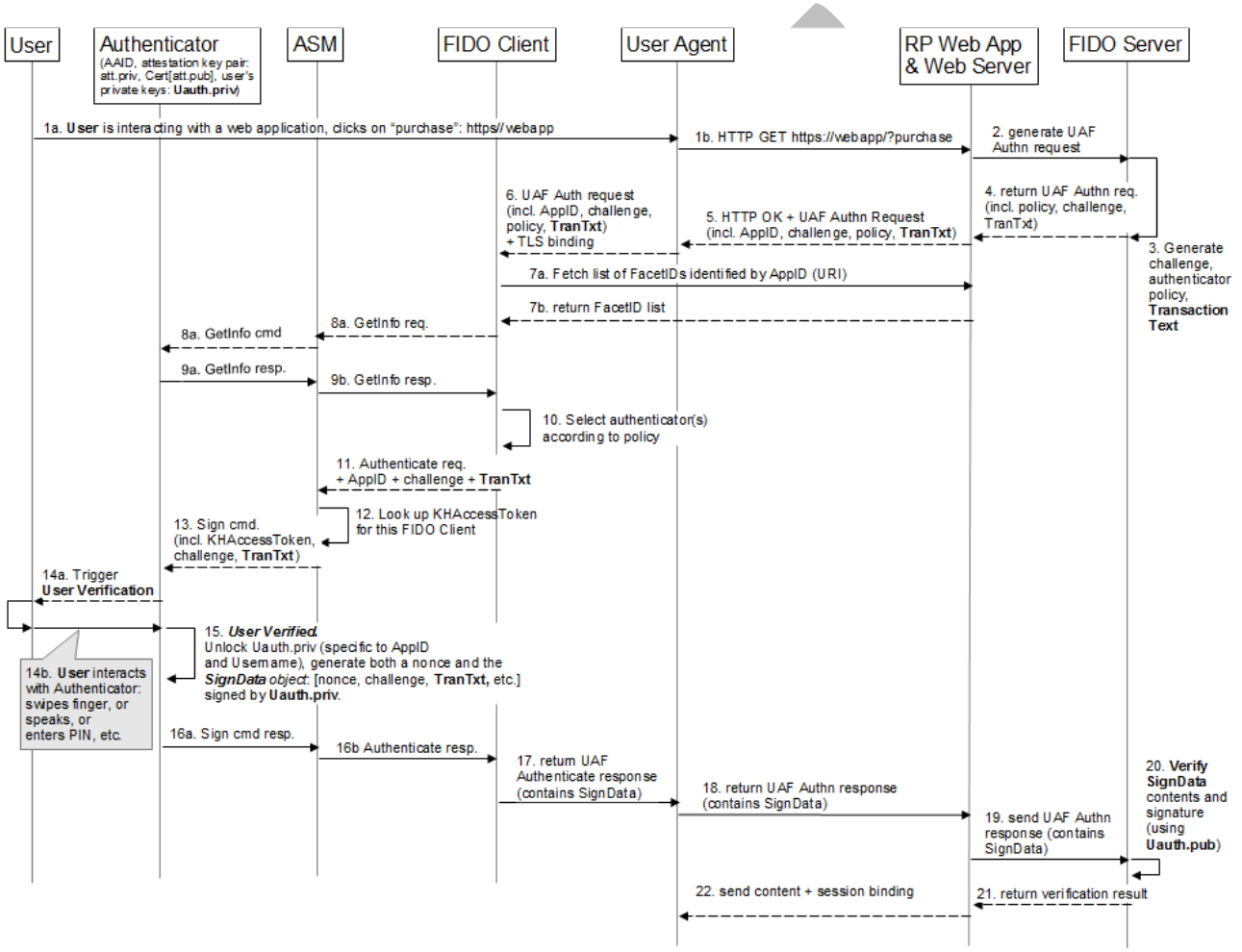

Figure 6 - UAF Authentication Sequence Diagram

This overall scenario will vary slightly depending upon the type of FIDO UAF Authenticator being employed. Some authenticators may sample biometric data such as a face image, fingerprint, or voice print. Others will require a PIN or local authenticator-specific passphrase entry. Still others may simply be a hardware bearer authenticator.

2.1.4.6 FIDO 2nd Factor Authentication (U2F)

Universal 2nd Factor (U2F) is an open authentication standard that strengthens and simplifies two-factor authentication (2FA) using specialized Universal Serial Bus (USB) or near-field communication (NFC) devices

based on similar security technology found in smart cards. Initially the standard was developed by Google and Yubico with contributions from NXP Semiconductors, but now hosted by the FIDO Alliance.

The FIDO U2F protocol enables relying parties to offer a strong cryptographic 2nd factor option for end user security. The relying party's dependence on passwords is reduced. The password can even be simplified to a four digit PIN. End users carry a single U2F device which works with any relying party supporting the protocol. The user gets the convenience of a single 'keychain' device and convenient security **[U2F2017]**. FIDO U2F allows online services to augment the security of their existing password infrastructure by adding a strong second factor to user login. The user logs in with a username and password as before. The service can also prompt the user to present a second factor device at any time it chooses. During registration and authentication, the user presents the second factor by simply pressing a button on a USB device or tapping over NFC or BLE. The user can use their FIDO U2F device across all online services that support the protocol leveraging built–in support in web browsers.

The U2F eco-system is designed to provide strong authentication for users on the web while preserving the user's privacy. The user carries a 'U2F device' as a second factor. When the user registers the U2F device at an account at a particular origin the device creates a new key pair usable only at that origin and gives the origin the public key to associate with the account. When the user authenticates to the origin, in addition to username and password, the origin can check whether the user has the U2F device by verifying a signature created by the device. The user is able to use the same device across multiple sites on the web - it thus serves as the user's physical web keychain with multiple (virtual) keys to various sites provisioned from one physical device. Using the open U2F standard, any origin will be able to use any browser (or OS) which has U2F support to talk to any U2F compliant device presented by the user to enable strong authentication **[U2F2017]**.

The U2F device can be embodied in various form factors, such as standalone USB devices, standalone Near Field Communication (NFC) device, standalone Bluetooth Low Energy (BLE) devices, built-on-board the user's client machine/mobile device as pure software or utilizing secured crypto capabilities. It is strongly preferable to have hardware backed security, but it is not a requirement. However, as we shall see the protocol provides an attestation mechanism which allows the accepting online service or website to identify the class of device and either accept it or not depending on the particular site's policy. Figure 7, below presents the basic process flow of U2F:

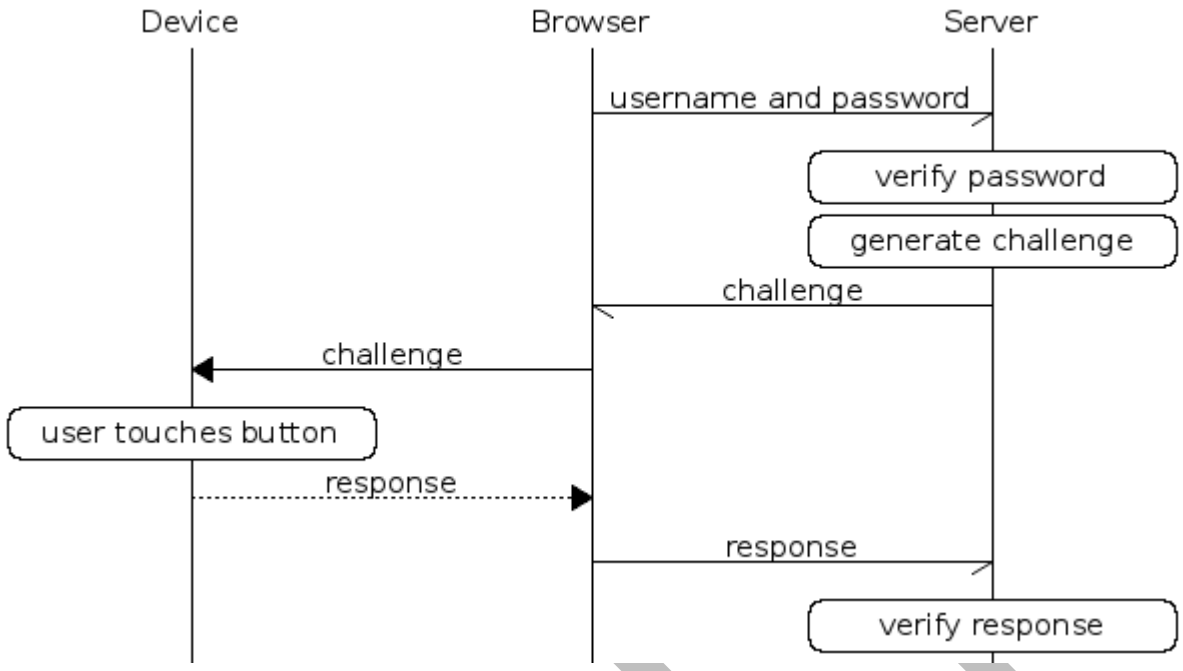

Figure 7 - U2F Basic Flow Diagram

U2F Registration

Below diagram shows the working of Registration with Attestation details.

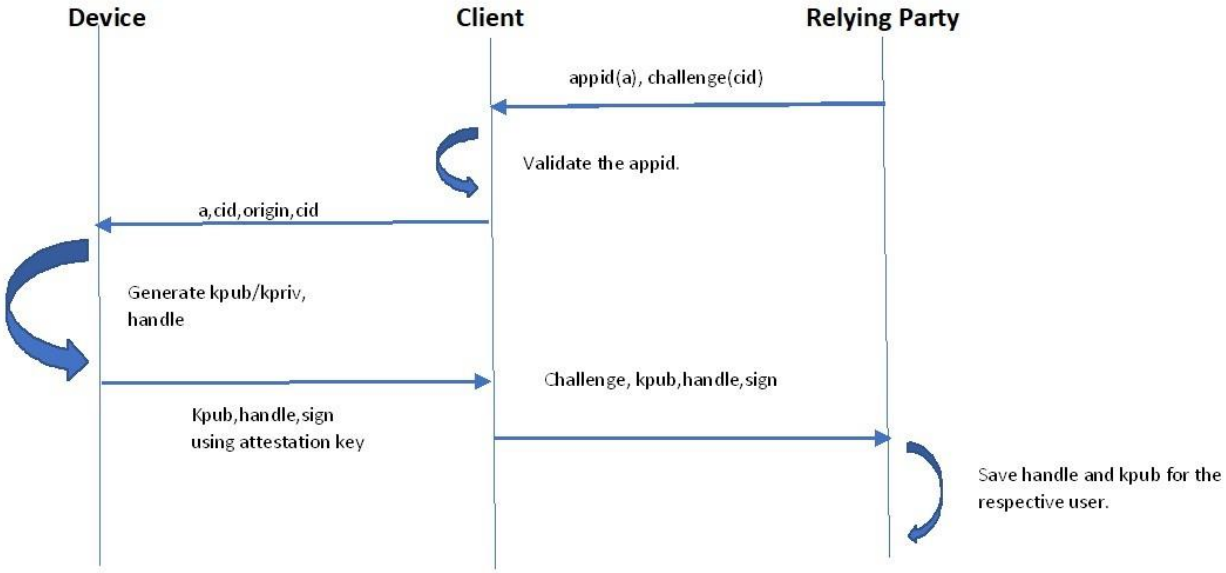

Figure 8 - U2F Registration

Each device class is provisioned with the attestation certificate (burnt into the device). The attestation public key can be found In the FIDO metadata server. The attestation private key is used to sign the registration response. Signing with the attestation private key gives a way to securely register the KPub/Key-handle with the relying party.

Flow of events:

- User clicks on Register, and the Relying party initiates the request with appid + challenge. Appid uniquely identifies the relying party and challenge is a unique random string.
- Client validates the appid before forwarding the request to device. If we look at Google's reference code, the validation involves checking the origin.
- Device receives the request and generates a pair of Kpub/Kpriv and key handle per RP per user on successful user presence check.
- User presence check is done on yubikeys by tapping on the u2f keys when it blinks.
- The attestation private key is used to generate the digital signature of the response containing challenge, key handle, public key of the device etc.
- When the relying party receives the response validates the response signature using attestation public key. Attestation public key can be found in FIDO Metadata server.
- The relying party saves the key handle and Kpub for the respective user.

U2F Authentication

During the registration process we saw that the Kpub was registered with the Relying party. Now in the authentication flow we will see how the Kpub will be used.

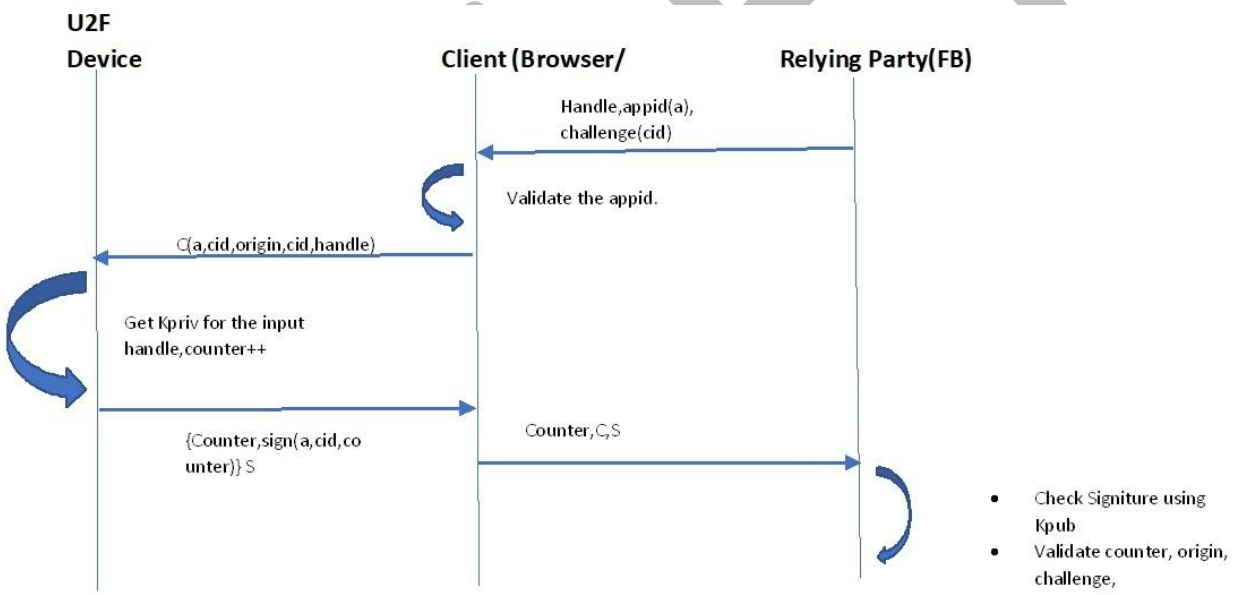

Figure 9 - U2F Authentication

Flow of events:

- User clicks on Login and enters the first factor. On successful authentication with the first factor; the second factor comes into picture.
- RP sends the Key handle/Challenge for the user trying to login to the client.
- The client-side application validates the appid and forwards it to the device.
- Device gets the Kpriv by looking at the key handle and increments the counter to mitigate replay attack on successful user presence check.
- Device sends back the response signed by KPriv.

- Relying party verifies the signature of the response using KPub the second factor and validates the counter, origin and challenge.

2.1.4.7 FIDO Web Authentication API (WebAuthN)

WebAuthn defines a standard web API that is being built into browsers and platforms to enable support for FIDO Authentication. Password-Free FIDO is a mechanism that allows users to log in into systems without the need of user and password credentials. FIDO was developed by companies that are part of the World Wide Web Consortium (WC3). The main idea is to improve the user experience and create a robust and secure mechanism for authenticating users in a system. The WebAuthn specification also defines a series of extensible points, such as the ability to add new attestation formats and the ability to add new extensions to the protocol and define their processing rules.

FIDO is using the pre-existent specifications: FIDO (Fast Identity Online), Universal 2nd Factor Authentication (U2F) and Universal Authentication Framework (UAF) for verifying user identities. To login into the system the user provides password information and as a second authentication mechanism biometrics or other mechanisms.

- **CTAP2** - allows the use of external authenticators (FIDO Security Keys, mobile devices) for authentication on FIDO2-enabled browsers and operating systems over USB, NFC, or BLE for a passwordless, second-factor or multi-factor authentication experience.
- **CTAP1** - The new name for FIDO U2F, CTAP1 allows the use of existing FIDO U2F devices (such as FIDO Security Keys) for authentication on FIDO2-enabled browsers and operating systems over USB, NFC, or BLE for a second-factor experience.

FIDO Authentication Flow

1. Initiate authentication with Relying Party
2. FIDO Server sends authentication challenge and preferences for the authenticators or credentials to be used
3. Authenticator performs user verification on device to signal the user's consent to authenticate with the service
4. The authenticator uses the service's origin to look up the private key for authentication and uses the private key to sign the challenge from the server. The server sends an authentication response: challenge + signature.
5. The server retrieves the public key for the user and validates the signature on the challenge.

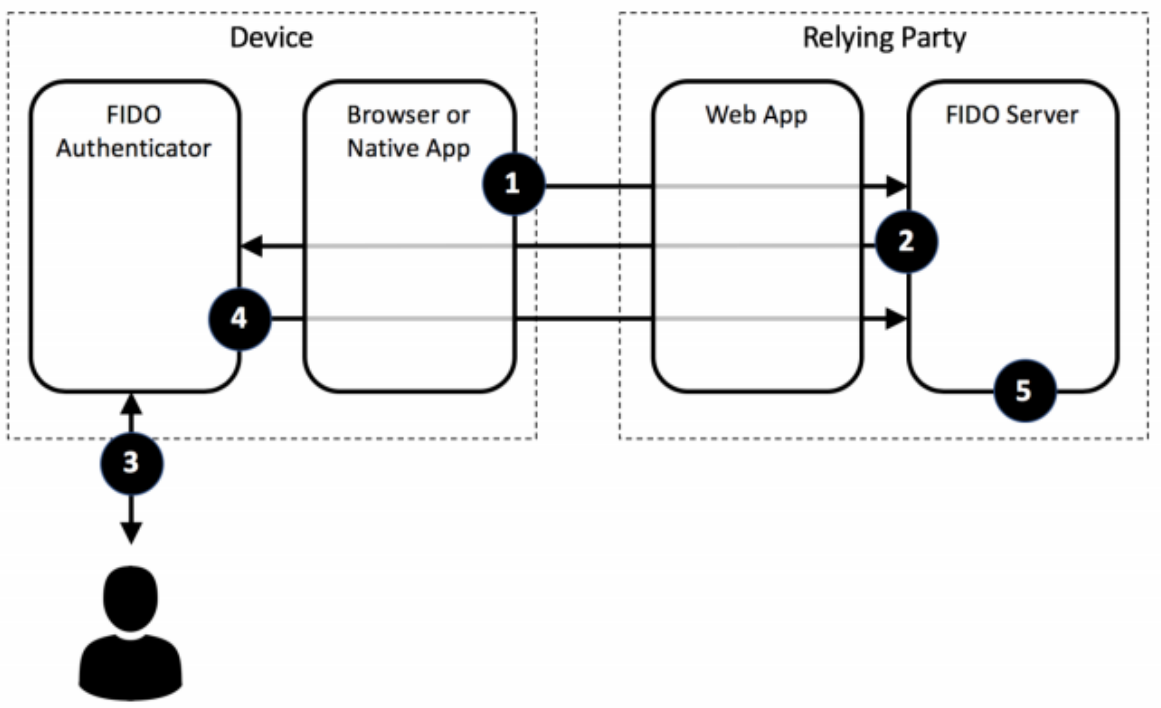

Figure 10 - FIDO Authentication Flow

2.1.4.8 Mobile Connect

Mobile Connect is the mobile operator-facilitated secure universal identity solution developed by the GSMA in collaboration with Mobile Operators. The GSMA represents the interests of mobile operators worldwide spanning more than 220 countries and unites nearly 800 of the world's mobile operators, as well as more than 230 companies in the broader mobile ecosystem. To-date there are more than 470 million active Mobile Connect users via over 70 operators covering more than 40 countries and reaching more than 3 billion people **[MOBILECONNECT]**. Itsme[1] is an example of mobile connect and provides to every Belgian citizen with a unique, secure mobile identity. More specifically, it offers a Belgian eID for identification purposes of all digital transactions.

Mobile Connect is a portfolio of mobile-enabled services that can be integrated into a Service Provider's application to support access to services provided by the Service Provider. Mobile Connect provides strong customer authentication, authorisation, and permissioned access to a User's identity and contextual network attributes.

Mobile Connect uses a distributed architecture in which each Mobile Operator deploys Mobile Connect services for its particular user base, but with all deployments abiding by a strict set of technical standards to ensure that from a Service Provider's perspective, the experience of consuming Mobile Connect services from any of the Mobile Operators is consistent

Mobile Connect is based upon the OpenID Connect protocol. It allows Users to be identified by their MSISDN (or a related Pseudonymous Customer Reference) and to be authenticated securely via their mobile device with the SIM providing security. Mobile Connect defines two profiles of OIDC to support

[1] https://www.itsme.be/en

Device-Initiated and Server-Initiated requests for authentication, authorisation or permissioned access to User attributes.
The serving Mobile Operator supports and selects an appropriate Authenticator to present the authentication and authorisation requests to the User on their mobile device to which the User responds. The Authenticator may also be used to seek User consent for the serving Operator to share or validate User attributes with the Service Provider. The Authenticator is selected based on Operator policy, device capability and the Level of Assurance required.

Mobile Connect also meets the eIDAS technical specification and interoperability requirements for integration with national ID as designed by EU Member States eIDAS Nodes in collaboration with the European Commission CEF project. An example reference architecture of eIDAS for the integration with Mobile Connect is shown in the following Figure.

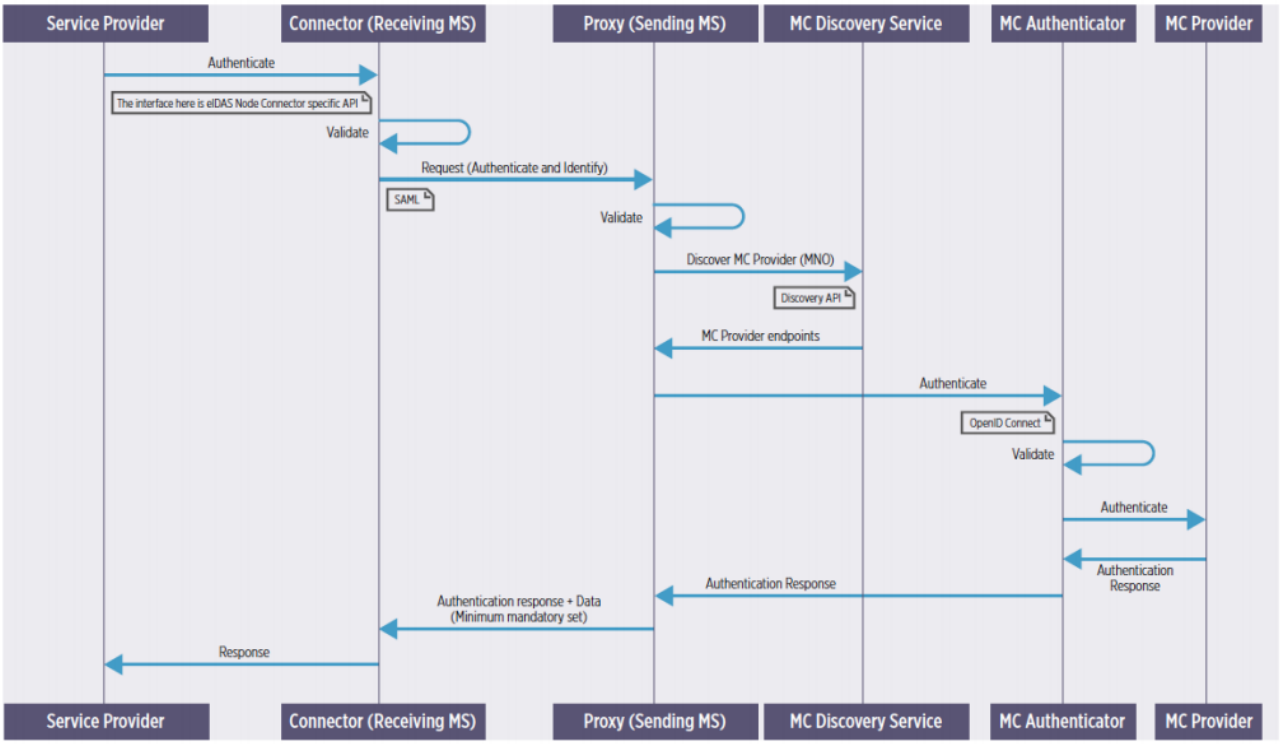

Figure 11 - Mobile Connect and eIDAS technical flow

2.2 Relation with other research projects

In this section we summarize some related research projects and Large Scale Pilots (LSP) that tried to solve the issue of interoperability and cross-border identification and authentication. The most known are listed below:

2.2.1 STORK

The goal of Secure idenTity acrOss boRders linked (STORK) project is to establish the cross-border recognition and authentication of e-IDs issued by other member states. The authorized use of e-IDs, secure access to work stations, and confidentiality, integrity and availability of personal data are the major challenges in this area. STORK aims at implementing an EU-wide interoperable system for recognition of eID and authentication that will enable businesses, citizens and government employees to use their

national electronic identities in any Member State. It's pilot trans-border e-government identity services and learn from practice on how to roll out such services, and to experience what benefits and challenges an EU-wide interoperability system for recognition of eID will bring.

The STORK interoperable solution for electronic identity (eID) is based on a distributed architecture that will pave the way towards the full integration of EU e-services while taking account of specifications and infrastructures currently existing in EU Member States **[STORK2010]**. STORK 2.0 is a pilot based on the STORK project and carried out by 19 European Member States and 59 partners of different types, such as governmental institutions, banks or universities. The initiative was planned with the aim of being helpful in the preparation of the eIDAS regulation. It is based on SAMLSTORK, which uses SAML extension capabilities to introduce new attributes and custom information. These modifications and the security restrictions make STORK incompatible with other standard SAML federations.

2.2.2 epSOS / eHDSI

Smart Open Services for European Patients (epSOS) is the main European electronic Health (eHealth) interoperability project co-funded by the European Commission and the partners. It focuses on improving medical treatment of citizens while abroad by providing health professionals with the necessary patient data **[EPSOS]**. epSOS aims to change this by ensuring standards for the exchange of medical information, subject to patient consent. epSOS aims to design, build, and evaluate a service infrastructure that demonstrates cross-border interoperability between electronic health record systems in Europe.

In the epSOS project identity management is one of the essential tasks that are being addressed. The core principle of epSOS is to bridge existing national eHealth infrastructures instead of setting up a new, centralised European healthcare service network from scratch. epSOS is trying to find solutions which are compatible with the national regulations and concepts of the participating countries. In epSOS a patient is not an active user of the platform, the patient does not perform any authentication procedure and does not use any of the epSOS software (this is the main difference between epSOS and InteropEHRate), he simply presents his identity documents to the HCP. On the other hand, the attending epSOS health professional is authenticated within the health professional's home country **[EPSOS]** and uses epSOS Identification Service to discover a valid patient identifier from an ID assigning authority by providing identifiers and/or demographic data that are sufficient for patient identification. The implementation of the epSOS Identification Service is based on the HL7 V3 Identification Service standard (HL7 IS) and is an extension to the IHE profile XCPD "Cross-Community Patient Discovery" **[IHE ITI TF-1]**.

Results of epSOS project have been used in its successor project called eHealth Digital Service Infrastructure (eHDSI or eHealth DSI). This project's objective is the initial deployment and operation of services for cross-border health data exchange (Patient Summary and Prescriptions) under the Connecting Europe Facility (CEF).

Despite InteropEHRate and epSOS/eHDSI projects are both focused on cross border health data exchange, the context in which they operate is very different, especially for what concerns authentication: in epSOS there is no authentication mechanism for the citizen, there is no app given to the citizen and the only user that performs an electronic authentication is the HCP, by using proprietary authentication mechanism provided by his country.

2.2.3 e-SENS

e-SENS (Electronic Simple European Networked Services) is aiming to consolidate and solidify the work done, to industrialize the solutions and to extend their potential to more and different domains **[ESENS2017]**. e-SENS focuses strongly on core building blocks such as eID, e-Documents, eDelivery, semantics and e-Signatures across the different LSP domains. The solutions will be based on already existing systems in Member States, and so no changes will be needed at national level.

e-SENS has been formed to consolidate and solidify the work done in previous LSP projects, and to extend these solutions to new domains. They will be tested for scalability and the ability to be reused in a number of domains. These building blocks aim to provide the foundation for the platform of "core services" for the eGovernment, cross-border, digital infrastructure foreseen in Regulation (EU) No 1316/ 2013 for establishing the Connecting Europe Facility (CEF).

3 INTEROPEHRATE SPECIFICATION FOR IDENTITY MANAGEMENT

The purpose of this section is to show how the InteropEHRate project will handle IDM and authentication mechanisms, for all the protocols and use cases. The following subsections include the crypto models for IDM and authentication for all the communication channels and involved applications. 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 12. More specifically, 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).

Identity management and authentication is done in most scenarios through Certificates acquired from a trusted CA under the same hierarchical root of trust. More specifically, in the context of D2D, demographic data are exchanged, while two variants of IDM are specified, a simple one that is coherent with the current procedures on health care institutes and a more research-oriented one that eliminates the need of ID-card and handwritten signature. In the context of R2D Access and in the second variant of RDS scenario an eIDAS-based solution is specified to achieve authentication and cross-border interoperability. In R2D Backup and Emergency, a simple credential-based login and an attribute-based access control mechanism (ABAC) is specified for IDM and authentication/authorization purposes respectively.

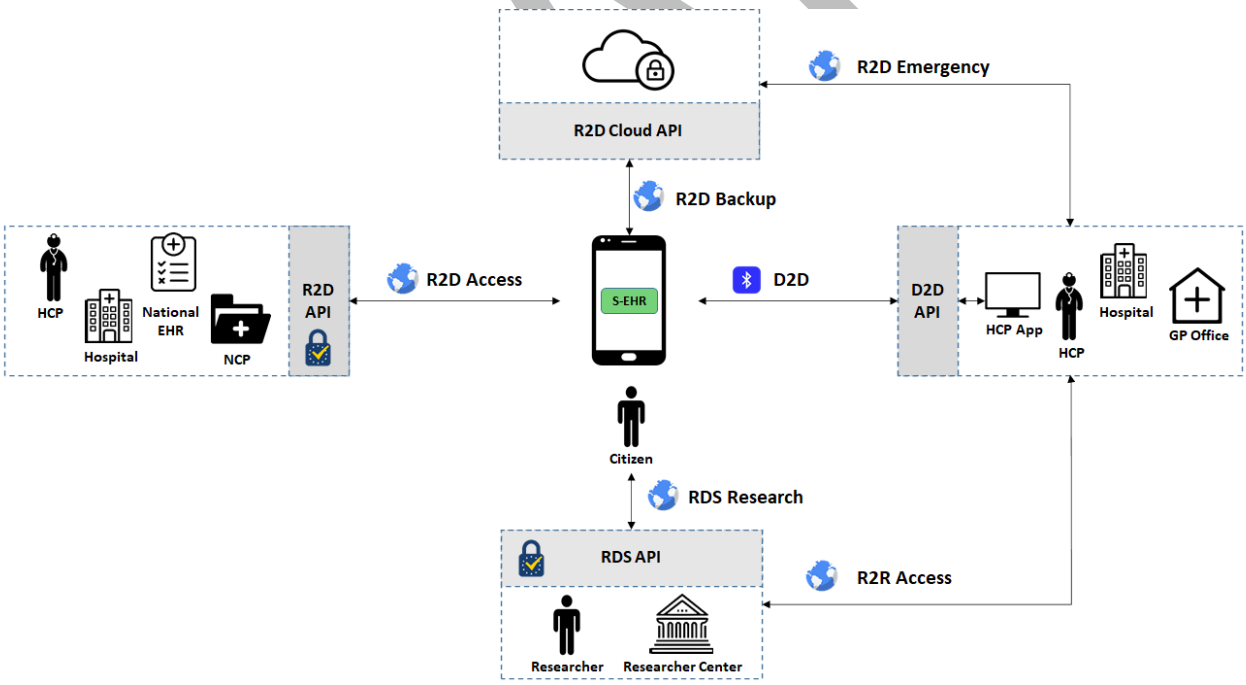

Figure 12 – InteropEHRate protocols

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. After the connection is established the HCP app and the S-EHR app send his/her demographic data for identification purposes (helloSEHR and helloHCP), while 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. These steps 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 section 3.11 of **[D3.6]** summarises all the security common remote APIs including the interaction with the CA and eIDAS in order to retrieve the necessary certificates, certificate chain and validate a certificate. In a nutshell, the eIDAS returned attributes (from the R2D Access protocol) during the identification phase are used as a valid input for the certificate generation **[D3.6]**.

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]**. As aforementioned, in the D2D protocol, we have two variants introduced in the first version of the deliverable:

1. In the first variant, the identification 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 certificates that correspond to the software signatures are acquired and verified from a trusted CA. As already referred to **[D3.6]**, different CAs are part of an hierarchical root of trust (under the same root CA) to achieve a cross border IDM. This first variant is simpler and feasible, and can be used immediately after the end of the project.
2. In the second variant, which will be used in the future, the identification will be done with hardware-based signatures (Qualified) from both parties for legal binding, while the demonstration of the paper-based ID-card is omitted. This variant was specified for experimentation reasons and not for demonstration purposes, during the duration of the project by replacing handwritten signatures to electronic signatures with legal binding under the eIDAS regulation.

The security models defined below will be the same independent of the software or hardware -based certificates and signatures. In addition the exchange of demographic data (name, surname, date of birth, place of birth) are not depicted since they are exchanged prior to any security operation and are plain messages. In the D2D protocol, we have two principals the $S-EHR\ App$ and the $HCP\ App$ that agree publicly on an element g that generates a multiplicative group G. The group G is a subgroup ofZ_p^* of prime order q, p is a large prime and $g$is a generator of the group Z_p^* of order m. Typical sizes in use today are 1024 bits for the length of p and 160 bits for the length of q. The two principals select random values, r_A and r_B respectively, in the range between 1 and the order of G. $S-EHR\ App$ calculates $t_A = g^{r_A}$and

$HCP\ App$ calculates $t_B = g^{r_B}$ and they exchange these values (public keys) as included in the corresponding Certificates. In order to generate the Certificates, both parties share their public keys to the CA, in order to issue the Certificate (i.e. C_A, C_B). 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}). It has also be note the HCP's Certificate C_B, should include the hospital name as an attribute for identification purposes of the HO. We assume that the HCP provides a document (proof of possession) that proves the HO he/she is working to, and the CA is responsible to check the validity of this document. However this process is out of the scope of the protocol. Each party can verify the Certificate signature (when it is necessary) with the CA's Certificate. These message exchanges are included in the helloSEHR and helloHCP invoked methods depicted in Figure 14. In the model we omit the steps of the encryption mechanisms and consent management, since they are part of other specification deliverables **[D3.6]** and **[D3.8]**, and we will focus only on identity management.

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.σ_B). This, apart from the integrity and authenticity, will be used for identification in a latter step of the protocol. 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. Verfunction), while the Nonce (i.e. N) is used for freshness. In the last message the consent (i.e. Con) is requested form the S-EHR app, however, more details will be provided in **[D3.8]**. 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. In addition, the S-EHR, when connected to the Internet, will have the ability (optionally) to validate the acquired certificates obtaining the revocation status by utilizing the OCSP protocol. Figure 13 below depicts the described security model.

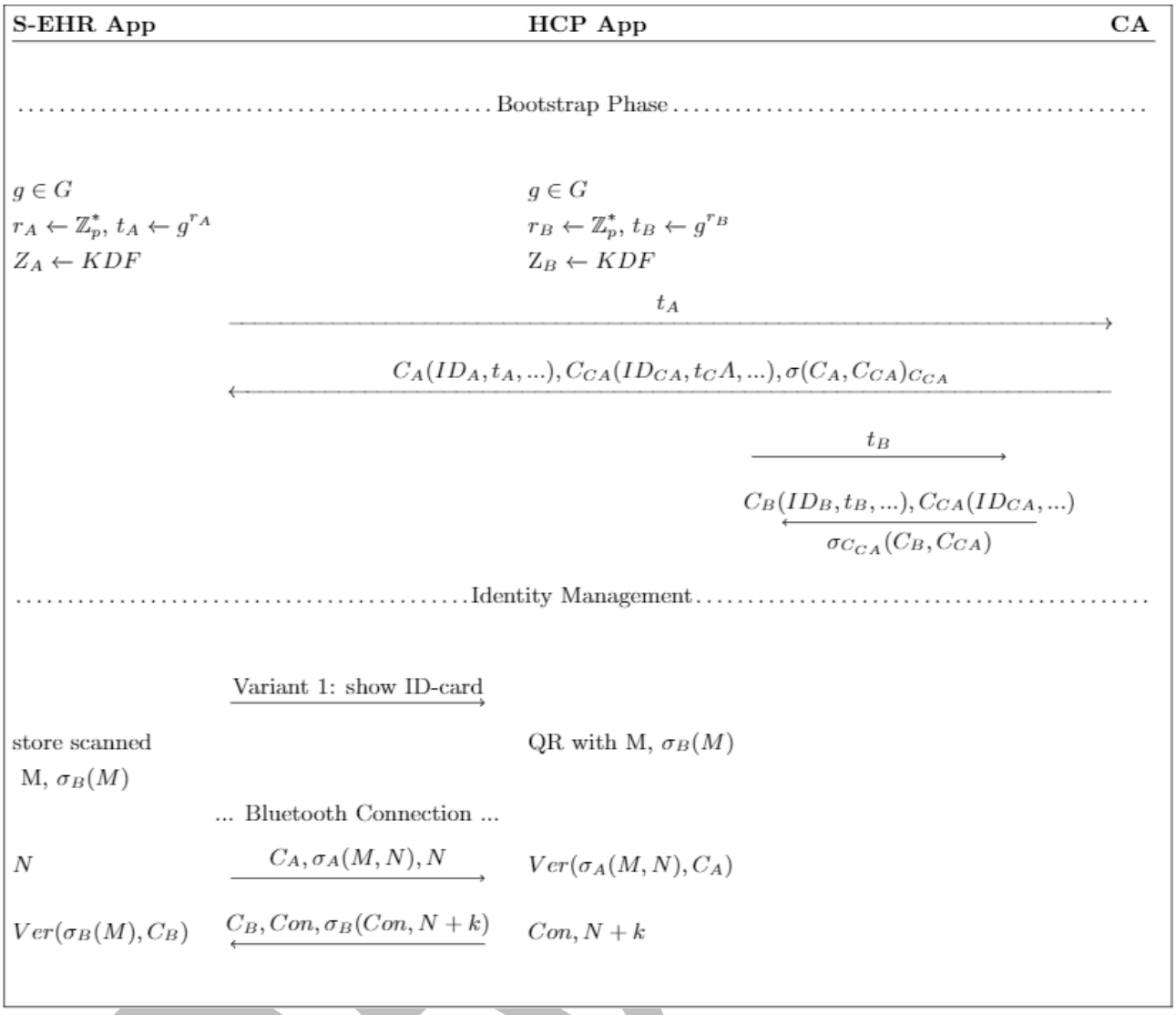

Figure 13 – D2D crypto-model

The conceptual sequence diagram that provides a high-level overview of the D2D is depicted in Figure 14, where the demographic identification data are exchanged. Following a detailed description of the sequence diagram of D2D for identification purposes:

- **Step 1**: The S-EHR app invokes this operation for getting the Practitioner's details (i.e., Healthcare Organization), for identifying whether the identity is valid or not.
- **Steps 2-3**: As soon as the decision has been made about the Healthcare Organization identity, this operation is invoked by the HCP app for getting the decision from the side of the S-EHR app, regarding whether the provided Health Organization identity is approved or not. This operation will return, one of the two following options:
    - **Step 2**: The demographic data of the S-EHR app owner (helloHCPApp) in the form of an object (Patient).
    - **Step 3**: A connection closure message (closeConnection) in the form of a String, indicating that the Health Organization identity was not approved, hence the connection will be closed.

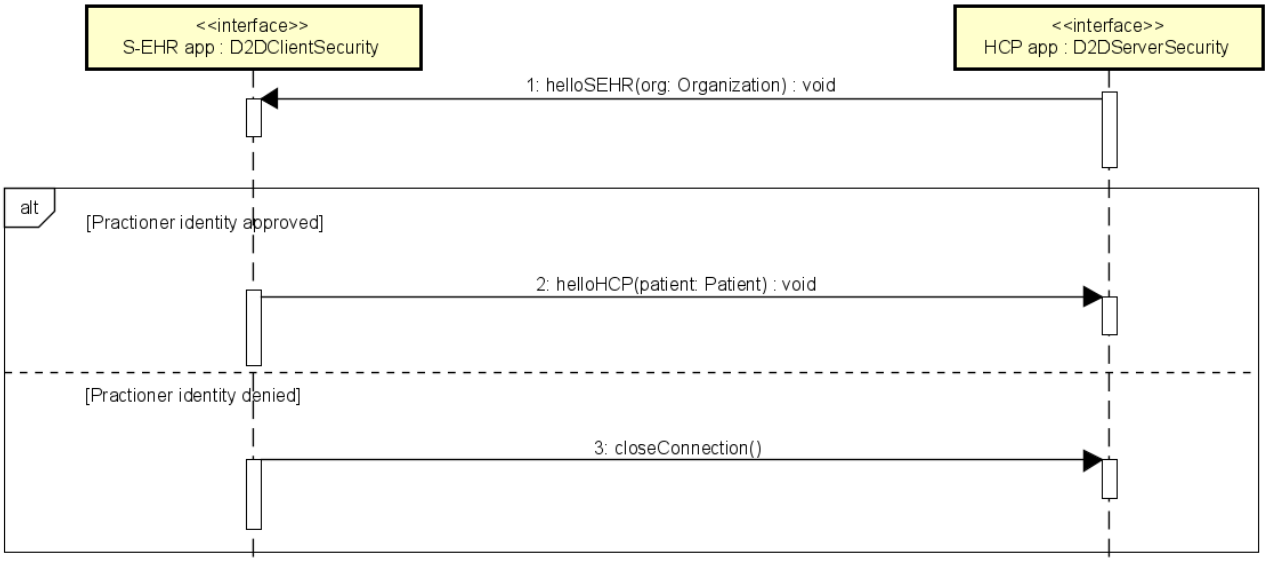

Figure 14 – D2D sequence diagram

3.2 D2D Security APIs

This section includes the APIs for the IDM and authentication mechanism in the D2D protocol. The S-EHR App, immediately after the Bluetooth connection is established, sends to the HCP App his/her demographic data for identification purposes. A detailed UML class diagram that demonstrated the involved interfaces of the D2D protocol is Figure 14 of **[D4.3]**. The security interfaces are the D2DServerSecurity and the D2DClientSecurity that contain all the operations for letting the HCP app and the S-EHR app establish a secure Bluetooth Connection. Appendix A also includes the structure of the exchanged Bluetooth messages.

3.2.1 S-EHR App Security APIs

Operation helloSEHR

Name	**helloSEHR**
Description	This method is invoked by the HCP App to send an instance of Organization (FHIR resource). The operation is for getting the Practitioner's details (i.e., Healthcare Organization), for identifying whether the identity is valid or not.
Arguments	• Organization org
Return Value	• void
Exceptions	• Security exceptions related to the Bluetooth connection • Network exceptions related to Bluetooth connection failure
Preconditions	• Bluetooth pairing has taken place

3.2.2 HCP App Security APIs

Operation helloHCPApp

Name	helloHCPApp
Description	This method is invoked by the S-EHR to send an instance of Patient. The operation is for getting the decision from the side of the S-EHR app, when the provided Health Organization identity is approved. The operation returns the demographic data of the S-EHR app owner (helloHCPApp) in the form of an object (Patient).
Arguments	• Patient patient
Return Value	• void
Exceptions	• Security exceptions related to the Bluetooth connection • Network exceptions related to Bluetooth connection failure
Preconditions	• Bluetooth pairing has taken place

3.3 R2D Access Security Architecture and Models

As already written previously, prior to any security operation a bootstrap phase is necessary in order for all the participants in the protocol to acquire the necessary elements. R2D Access leverages an eIDAS-based architecture for cross-border identification/authentication of the citizen supporting the trust services and electronic identification, as defined by the current eIDAS framework. In the R2D Access, the involved parties have to acquire the necessary Certificates in order to verify the integrity and authenticity of the assertion response. The Certificates (i.e. C_A, C_L, C_E) must be retrieved in order to achieve the cross-border authentication and encrypted communication. Figure 15 depicts the R2D Access crypto model.

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. The eIDAS interoperability framework allows for cross-border identification and authentication processes through the exchange of SAML 2.0 messages, including personal and technical attributes. The SAML Response (the token) is digitally signed 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.

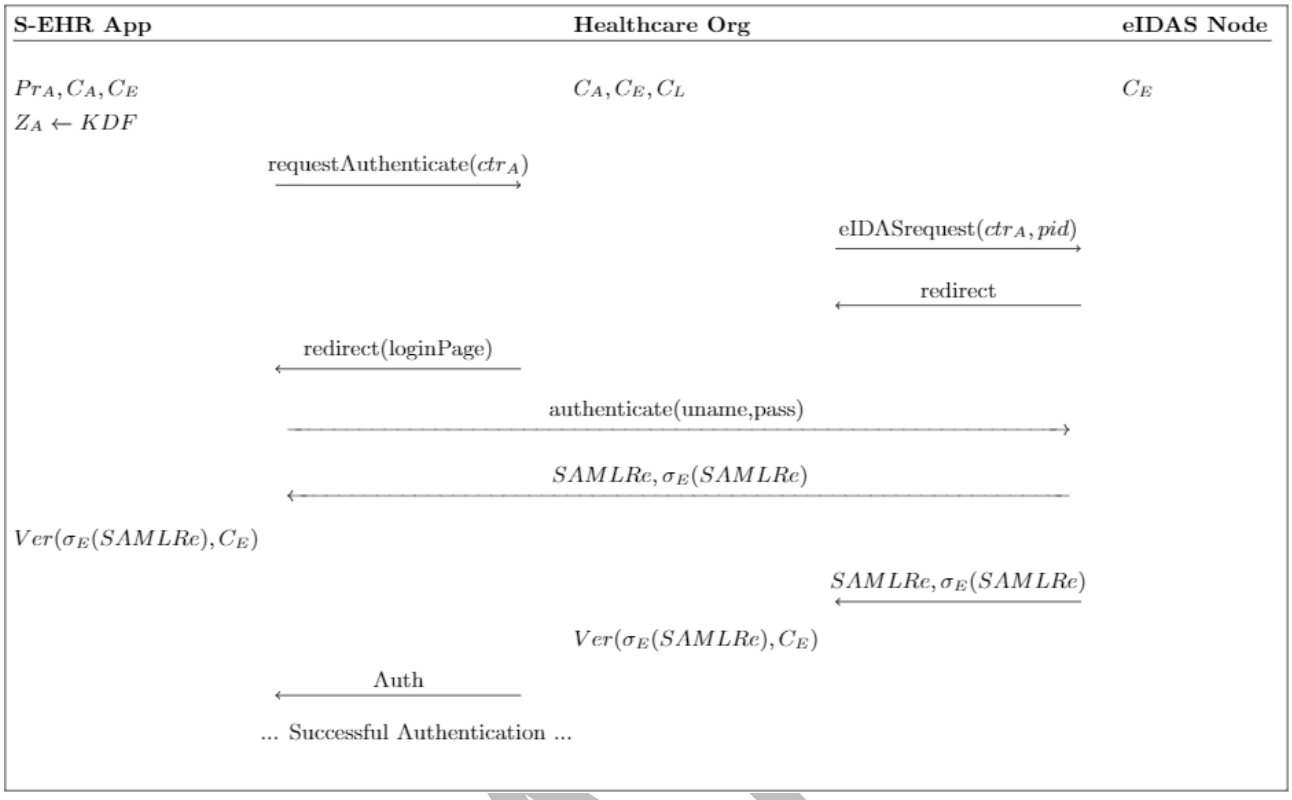

Figure 15 – R2D Access crypto-model

The conceptual sequence diagram that provides a high-level overview of the R2D Access is provided in the **[D3.3]**, where the detailed description of the defined steps is analysed. The updated version is presented in Figure 16. Even though the interfaces are not depicted in the following figure we will refer to them for easier reference of the reader. 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 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 a 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 at distance (by means of the Internet) any DICOM study of the subject citizen referred by FHIR resources imported by means of R2DAccess or D2D. The interface R2DAccessDICOM is optional, i.e. a HCO System may offer the R2DAccess without offering R2DAccessDICOM **[D2.6]**. The workflow of actions of the login Interface, including the exposed APIs of the Trusted Proxy Server, that acts as a "bridge" between the Healthcare EHR and the eIDAS infrastructure for providing a connection point to an eIDAS (proxy-based) node depending on specific Healthcare EHR requirements for certifying a requesting user. This solution simplifies the secure interaction of any Healthcare EHR, regardless of the programming language used for the EHR implementation, and the secure exchange of all required information for the eIDAS-based user certification and authentication. In a nutshell, the Trusted Proxy Server: a) enables Healthcare EHR entities to communicate with the eIDAS infrastructure without using SAML 2.0 - a time consuming development process, and b) supports platform independent EHRs since the communication is implementation-agnostic as it is based on REST APIs. The Trusted Proxy Server, as its name suggests, acts as a proxy between the

actual Healthcare EHR and the eIDAS infrastructure. It handles all the interactions with the eIDAS, and requires from the Healthcare EHR to receive the Country and the UserAttributes. At the end of an eIDAS authentication process, it receives the identification attributes from the eIDAS Node and generates a signed JWT token containing those attributes. Finally, the token is sent back to the Healthcare EMR for the subsequent user authentication which, if successful, will result in the transmission of the healthcare records back to the user. The Healthcare organisation is responsible for deploying this Trusted Proxy Server. Following a detailed description of the sequence diagram of R2D Access identification and authentication:

- **Step 1**: We assume an (already) eIDAS registered citizen that tries to get access to the R2D Access Server in order to retrieve his/her healthcare records for the first time. The citizen from country A with a S-EHR App requests to be authenticated through eIDAS infrastructure by selecting his/her country of origin. The request is redirected to the Trusted Proxy Server, a service that runs inside the healthcare organisation of country B. Then, an eIDAS request is performed, where the eIDAS Connector redirects the request to the eIDAS node of the country of origin of the User (eIDAS Proxy) - in the case that the country of origin is different to the country where the request was made. Finally, the corresponding login page of his home country IdP is redirected to the citizen. The successful finalisation of this step returns an eIDAS token, however how this is obtained is not part of the protocol. Last but not least, It has to be noted here that part of the protocol is only the requestAuthenticate method, while the are steps and sub-steps are part of the InteropEHRate Framework.
- **Step 2**: The citizen inserts in the login page the necessary credentials and the IdP of his/her country checks the validity and returns an authentication token upon successful authentication. The output of the eIDAS authentication flow is an eIDAS SAML authentication response. This token is stored for authentication purposes of the citizen in the R2D Access protocol and in the second variant of RDS protocol. The citizen can request an authenticated service with this received token. In the R2D Access request his/her health data to be downloaded in the S-EHR App. If the procedure is successful, the user can then access and download the requested records provided by Healthcare EHR.

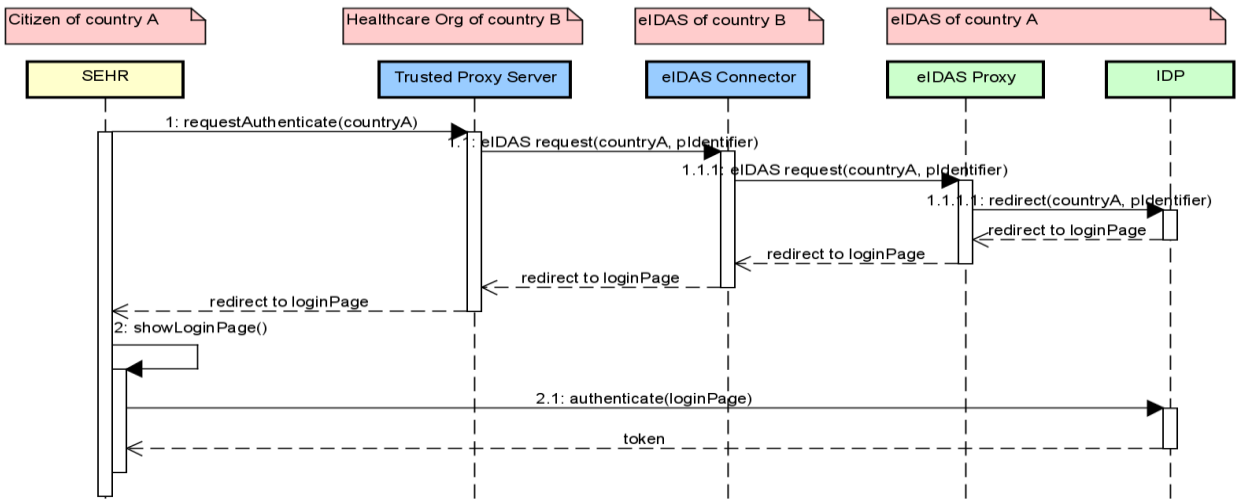

Figure 16 – R2D Access sequence diagram

3.4 R2D Access Security APIs

3.4.1 S-EHR App Security APIs

Operation

Name	EidasWebViewActivity
Description	This activity handles the redirection of the user to the eIDAS login page hosted by the Trusted Proxy Server. The Trusted Proxy Server is responsible for managing all the communication between the eIDAS infrastructure and the S-EHR App.
Arguments	• String login_url: The login url of the Trusted Proxy Server for authentication. The user is redirected to this url in order to add his/her eIDAS credentials.
Return Value	• String token: jwt token upon successful authentication with eIDAS and a failure message upon failed authentication.
Exceptions	• Exception
Preconditions	• Trusted Proxy Server is available.

Operation decode

Name	decode
Description	This method is responsible for decoding the eIDAS response details. The details include the
Arguments	• Context context: Android context. • String jwt: The token returns upon successful authentication.
Return Value	• ResponseDetails res
Exceptions	• InvalidKeySpecException
Preconditions	• Successful authentication of the user.

3.5 R2D Backup Security Architecture and Models

The purpose of the R2D Backup protocol is to allow citizens to securely back up their EHR in a remote repository (i.e. a S-EHR Cloud provider of their choice). The R2D Backup protocol defines a set of operations used for enabling (in a standardized way) the upload of encrypted health data on a S-EHR Cloud service along with the download of this encrypted health data to a S-EHR Mobile App from a S-EHR Cloud. Prior to any backup an authentication mechanism is needed for the S-EHR App. 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 identity and at the same time assure the integrity and authenticity of the consents (see also **D3.8**). Figure 17 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.

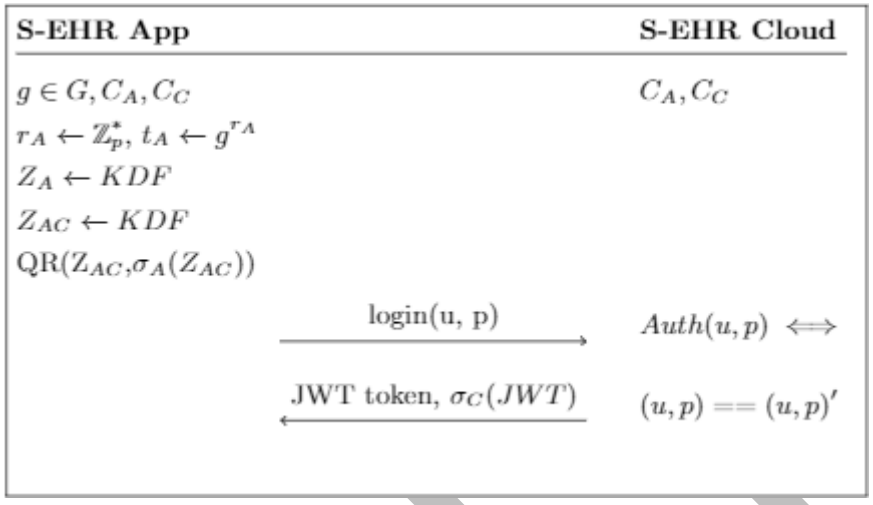

Figure 17 – R2D Backup crypto-model

The conceptual sequence diagram that provides a high-level overview of the R2D Backup is presented in Figure18. Prior to the emergency the citizen decides to use the optional S-EHR Cloud service. Through their S-EHR App, the citizen chooses their preferred S-EHR Cloud provider and creates an account. Upon user registration, S-EHR App tries to login the optional service using the same credentials used for registration. After the successful account's creation and login, the citizen then needs to agree on two consents. More details for the consent management will be provided in **[D3.8]**. Following a detailed description of the sequence diagram of R2D Backup identification and authentication:

- **Step 1**: The S-EHR App chooses the preferred S-EHR Cloud provider and creates an account to backup his/her data in encrypted form. The S-EHR App provides the credentials for authentication at the next step.
- **Step 2**: The S-EHR App tries to log in in the optional S-EHR Cloud service, by providing the selected credentials (e.g. uname, pass). Upon successful authentication a signed JWT token is returned back to the S-EHR App.

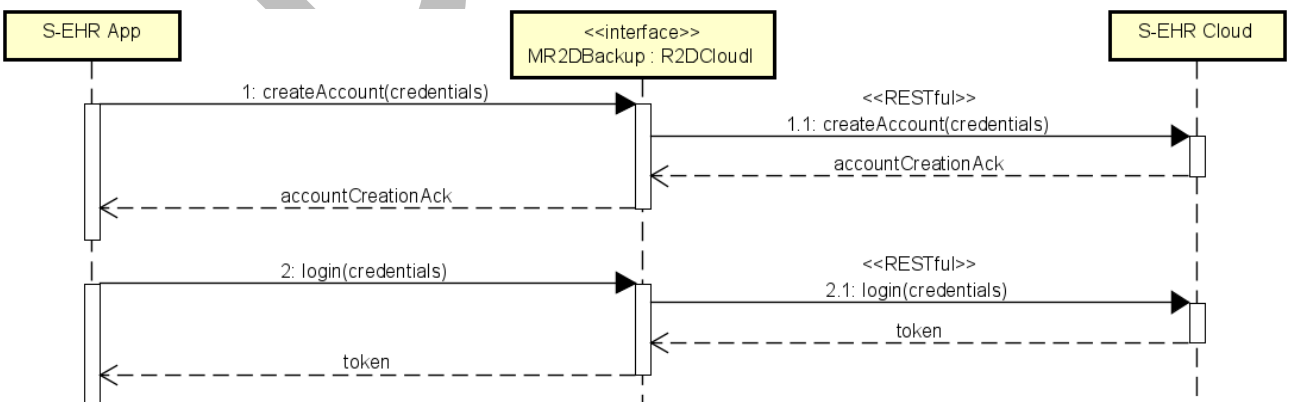

Figure 18 – R2D Backup sequence diagram

3.6 R2D Backup Security APIs

3.6.1 S-EHR Cloud Security APIs

Operation login

Name	Login
Description	Login of a citizen on the S-EHR Cloud. It is a POST request in the S-EHR-C endpoint http://[base url]/citizen/login
Arguments	• String username:password
Return Value	• String token: An authentication token ○ **200 Successful:** request was successfully processed. ○ **400 Bad request:** request could not be processed. ○ **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed.
Exceptions	• Missing argument: Authentication token
Preconditions	• The user should be registered in the S-EHR-C.

3.7 R2D Emergency Security Architecture and Models

The purpose of the R2D Emergency protocol is to allow authorized HCPs to gain access to the health data the citizen backed up on the S-EHR Cloud during an emergency situation. The R2D Emergency protocol defines a set of operations used for enabling (in a standardized way) the download of encrypted health data by an authorized HCP from a S-EHR Cloud service to an HCP App, along with the ability upload to the S-EHR Cloud of that specific citizen in need, new health data regarding that emergency, and reports at patient discharge, once the emergency is over. Figure 19 depicts the R2D Emergency crypto model. To this end, the HCP app will be able to acquire the symmetric key Z_{AC} and verify its authenticity and integrity (i.e.Ver) after scanning the QR code (generated in R2D Backup protocol). In addition, the HCP App needs to first authenticate itself with a Certificate (i.e.C_A) issued by a CA and containing custom attributes, including the Health Organisation (HO) and the profession, to the S-EHR Cloud (i.e.$requestaccess(C_A)$). 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 login is intercepted by an Attribute-based Access Control (ABAC) engine (e.g. the Authorization server) necessary for HCP authorization. We also assume that the Authorization serve receives a user's attributes from a trustworthy external source known as an attribute provider (AP). 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. Finally, the HCP App downloads the encrypted data.

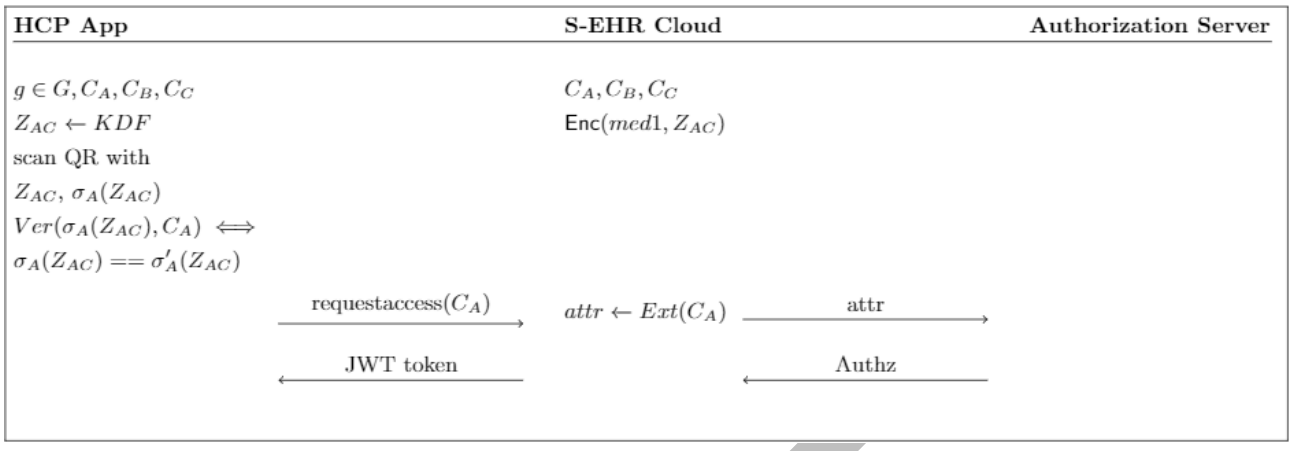

Figure 19 – R2D Emergency crypto-model

The conceptual sequence diagram that provides a high-level overview of the R2D Emergency authentication/authorization is presented in Figure 20. More details regarding the ABAC-based authorization will also be provided in **[D3.8]** as a deliverable responsible for the authorization aspects. Once the emergency occurs, the citizen is transferred to a healthcare facility. With the phone being unreachable the HCP that cures the citizen, uses their HCP app to connect to the S-EHR Cloud service that the patient uses in order to access their health data. The information regarding the S-EHR Cloud is collected from the QR-code provided by the patient, and scanned by the HCP. Upon successful authorization a JWT authorization token is returned to the user. Following a detailed description of the sequence diagram of R2D Emergency identification and authentication/authorization:

- **Step 1**: The HCP scans the QR-code of the patient in order to acquire the symmetric key for data decryption, and the emergency token necessary to connect to the cloud.
- **Step 2**: The HCP requests access to the R2D Cloud along with the emergency token and the attributes, the request is redirected to the Authorization server, where the decision whether the access is granted or not is returned to the HCP App. If the authorization decision is positive a signed JWT authorization token is returned.

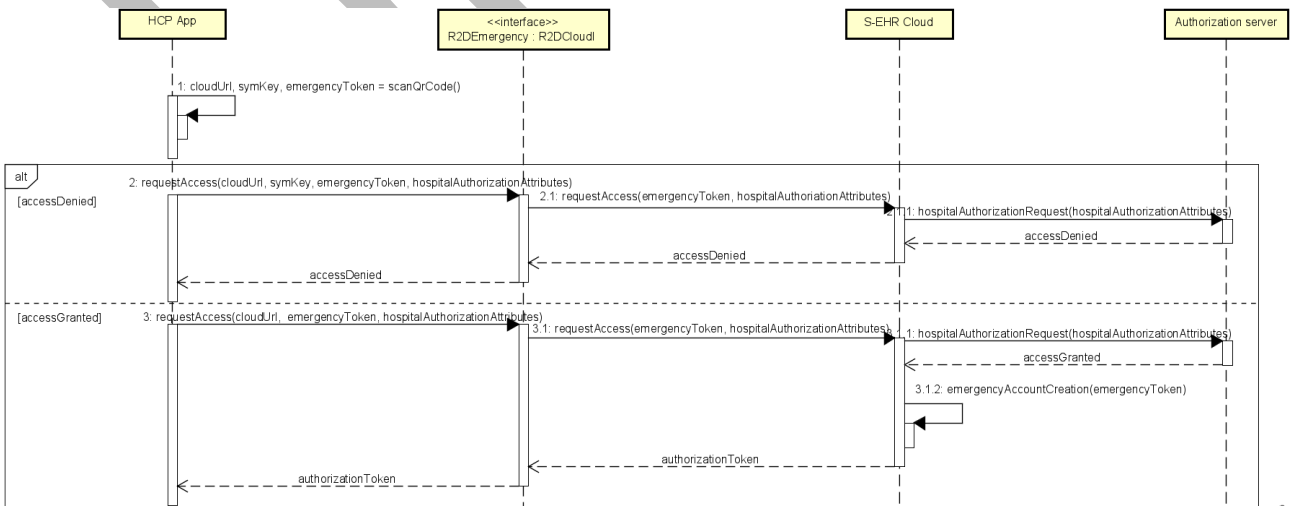

Figure 20 – R2D Emergency sequence diagram

3.8 R2D Emergency Security APIs

3.8.1 HCP Web App Security APIs

Operation requestaccess

Name	requestaccess
Description	HCP's request to access the citizen's bucket during an emergency. An account is created for the health care institution that cures the patient. 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 in a bucket dedicated for that specific emergency.
Arguments	• HCP authorization token: String • Health care institution attributes: String • HCP attributes: String
Return Value	• String token: Health care institution authentication token ○ **200 Successful:** request was successfully processed. ○ **400 Bad request:** request could not be processed. ○ **401 Not Authorized**: authorization is required for the interaction that was attempted. ○ **500 Internal Server Error**: server encountered an unexpected internal error, the request could not be processed.
Exceptions	• Missing argument: HCP authorization token • Missing argument: Health care institution attributes • Missing argument: HCP attributes
Preconditions	• The citizen should have agreed to share their health data with authorized HCPs during emergency situations. • 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.

3.9 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 **[D4.8]**. Two variants are supported on this protocol, one a) with pseudo-identities and another b) with pseudonyms. The second variant uses high-entropy pseudonyms. Entropy provides the measure of the uncertainty to identify the citizen that is participating in a study among a set of citizens. In this protocol the S-EHR App supports opt-in to a study and uploads anonymised data on the selected RRC (Reference Research Center). In addition, a trust-relation is already established between a Pseudonym Provider (PP), the eIDAS Node, the CA and the identity provider (IDP) with which the S-EHR App has been registered. There is no need to depict this in the security model, but the provided APIs are described in the following section. As a precondition to requesting the issuance of pseudonyms, the citizen must have acquired his/her certificate **[eHDSI2021]** and retrieve an anonymous SAML assertion that can later be used to anonymously authenticate him/herself to the PP as a legitimate entity. The issuance of this anonymous assertion can be supported by any IDP used to

check the identity of the citizen when requesting the issuance of a certificate, from the eIDAS, the CA, or the CA itself as part of the common criteria defined for the certificate policy management **[LABIOD2018]**. In our case since eIDAS is also used as the IDP we also consider the interaction between the CA and the eIDAS for verifying the ID of the citizens **[D3.6]**. Thus the CA upon request of a certificate issuance of a user interacts with eIDAS to check the ID of the citizens; if verification is successful the eIDAS SAML response contains also this anonymous token (currently supported by default by the standard eIDAS response data model). Overall, the issuance of the certificate and the anonymous token is a precondition. SAML assertions carry the following types of security claims **[Gisdakis 2013]**: a) authentication statements, b) authorization statements and c) 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 (see section 3.3) 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.

As described in the previous protocols in the bootstrap phase, S-EHR App, PP and RRC have already access to the needed Certificates (C_A, C_E, C_R, C_P), 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 and S-EHR App and RRC has run the key agreement phase to establish a shared key Z_{AR}(the same usein in Figure 13). Prior to any shared information, two consents are taking place, however more details will be provided in **[D3.8]**.

In the first variant the protocol the Principal Investigator (PI) of the RRC creates this pseudo-id (i.e. pid) and in the second one the PP creates the pseudonym (i.e. $pseu$) based on the Group-based signatures (i.e. $GBGen$), after the successful authentication of the S-EHR App, with the eIDAS SAML assertion T_{ID}. More specifically, the S-EHR App requests a pseudonym generation (in an anonymous way) from the PP without disclosing a real identity. Thereupon, the PP issues an authentication request designated for the eIDAS Node. This request is relayed by the S-EHR App. Consequently, the S-EHR App engages in an interactive authentication protocol with the eIDAS Node, based on TLS and their digital certificates. Upon successful authentication, the eIDAS Node issues an authentication response that contains a SAML assertion. However, this phase has already performed during the R2D Access protocol. The S-EHR App forwards the second part, the anonymous SAML assertion T_{ID} to the PP. Upon reception, the PP validates the authentication response and examines the eligibility of the S-EHR App with respect to the service. Accordingly, it grants or denies access. The complete protocol runs over Hypertext Transfer Protocol (HTTP). To authenticate both the eIDAS Node and the PP, one-way TLS authentication that additionally ensures the confidentiality and integrity of communications is leveraged. It has to be noted here that more than one pseudonym can be generated and used (i.e. $pseu_1, \ldots, pseu_n$). Each pseudonym is represented by public - private pairs. The public one is a X.509 certificate C_{AP}, while the private ones associated with this certificate are many $Pr_1, \ldots, Pr_n$. The S-EHR App with more than one pseudonym can achieve unconditional anonymity compared with the first variant, since there is no possible way to link different signatures. One of the two pidor $pseu$based on the variant will be used to anonymize the data (i.e. An-anonymous signing) and the encrypted (i.e. Enc) with the agreed key is transferred to the RRC. The RRC decrypts (i.e. Dec) with the same key the data, verifies the anonymous signature (currently this is not

depicted for space reasons - the certificate is assumed that is sent along with the anonymised data) and retrieves the anonymised data for further process and survey.

For liability attribution, a mechanism to trace a pseudonym back to the S-EHR App long-term identifier, is also provided. In this context, RRC might additionally request pseudonym resolution and notification of a patient in case of emergency like a newly discovered diagnosis (or other relevant medical assumption) uncovered by the research center: a) in the first variant the PI does the mapping (i.e. $Map(pid)$) on the real identity, namely the long-term id L_{ID}, while b) in the second valiant this mapping is done on the PP by requesting information form the IDP and then returned to the RRC in order to further notify the Citizen. More specifically, the RRC requests the transient identifier of the SAML assertion from the PP for which the pseudonym was issued. The PP responds with the corresponding transient identifier T_{ID}. The RRC then provides the eIDAS Node with the transient identifier and the eIDAS Node provides the RRC with the long-term identity (of the S-EHR App) and the list of all the transient identifiers for which it has issued assertions.

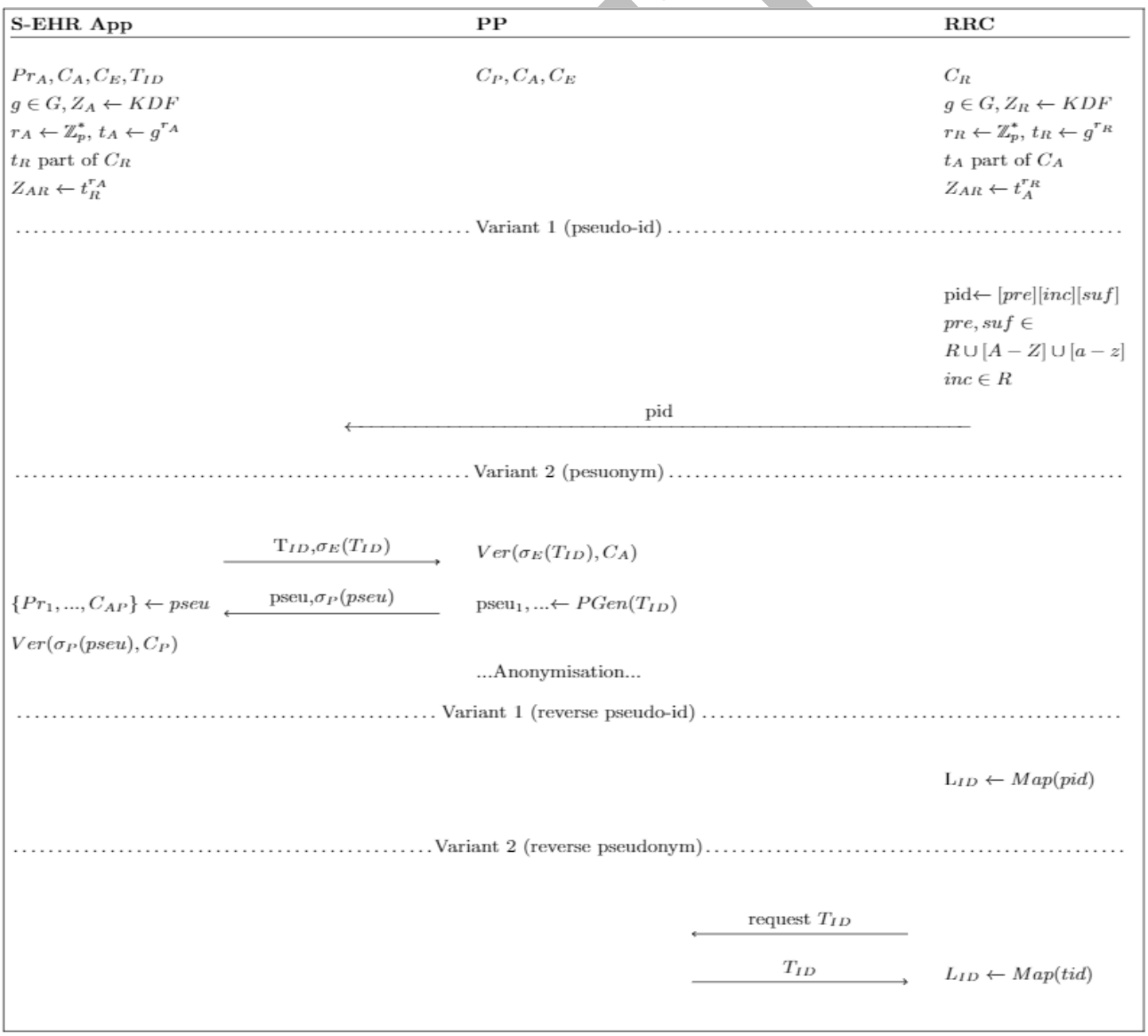

Figure 21 – RDS crypto-model

The conceptual sequence diagram that provides a high-level overview of the identification mechanism in the second variant of RDS protocol is presented in Figure 22 and the pseudonym resolution in Figure 23. Following a detailed description of the sequence diagram of the second variant of RDS identification and authentication of the S-EHR App to the PP:

- **Step 1**: The S-EHR app holds an authorization assertion. We assume this step as a prerequisite and out of the scope of the actual protocol. As aforementioned, the issuance of this anonymous assertion token can be supported by any IDP used to check the identity of the citizen when requesting the issuance of a certificate, from the eIDAS to the CA. In our case since eIDAS is also used as the IDP during the R2D Access we also consider the token is already acquired. Figure 16 represents thoroughly all the required message exchanges in order for the eIDAS-based assertion token to be retrieved. This token will be used to request a high-entropy pseudonym generation from the Pseudonym Provider. S-ERH app request from the IDP to map the signed assertion of the user with his/her public key; if the mapping is successful the access_condidtion_token is returned to the user necessary to request pseudonym form the Pseudonym Provider.
- **Step 2**: S-EHR app request from the Pseudonym provider to generate a high-entropy pseudonym; if the generation is successful the pseudonym is returned in the form of public/private keys to the S-EHR app.

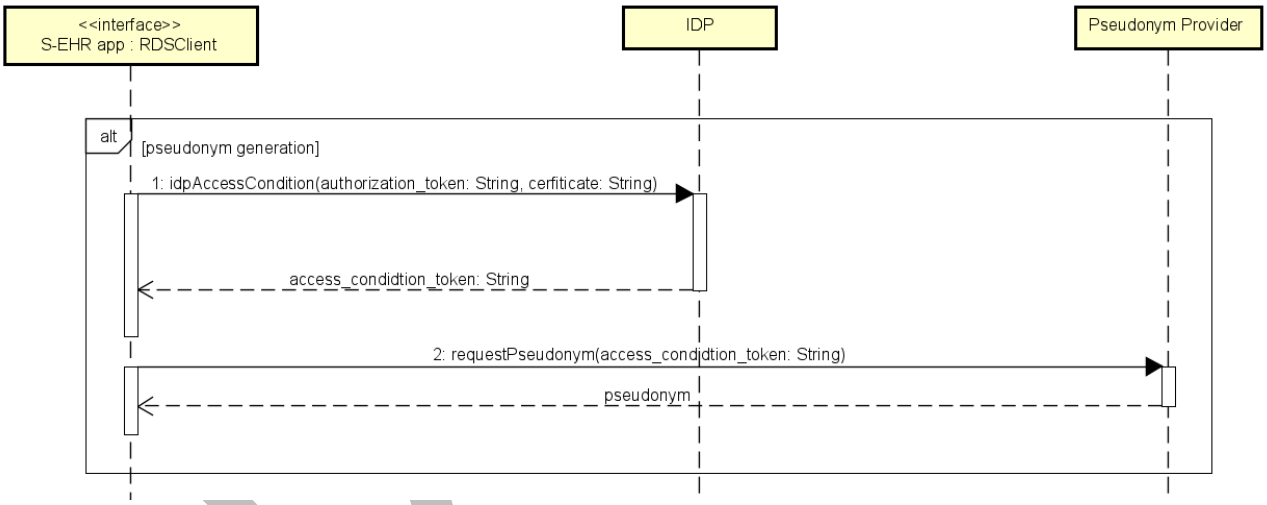

Figure 22 – RDS sequence diagram (pseudonym generation)

Following a detailed description of the sequence diagram of the second variant of RDS pseudonym resolution of the RRC to the PP:

- **Step 1**: In this step the RRC uses the certificate of the user to the Pseudonym Provider for the identity resolution of the patient for emergency purposes.
- **Step 2**: In this step the RRC uses the retrieved token from the first step to the IDP in order to retrieve another token containing information regarding the user certificate and the assertion.

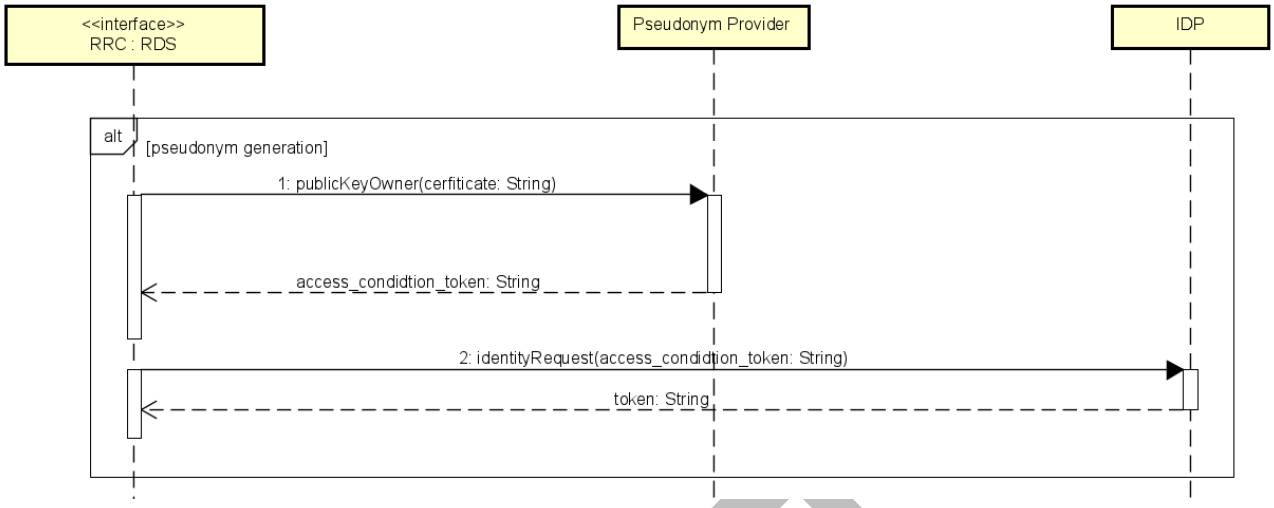

Figure 23 – RDS sequence diagram (pseudonym resolution)

3.10 RDS Security APIs

3.10.1 IDP APIs

Operation idpAccessCondition

Name	**idpAccessCondition**
Description	This functionality maps the signed assertion of the user with his public key in order to request for pseudonyms from the PP.
Arguments	• authorization token (a signed assertion): String • user certificate: String
Return Value	• access_condidtion token: String that includes an encrypted and signed message of the access condition (successful or not) of the mapping.
Exceptions	• N/A
Preconditions	• Already authenticated to eIDAS • Already register and acquire Certificate from the CA

Operation identityRequest

Name	**identityRequest**
Description	This functionality is used for resolution of the identity of the patient for emergency purposes.
Arguments	• access_condidtion token: String that includes an encrypted and signed message of the access condition (successful or not) of the mapping.
Return Value	• token String: a signed token that includes information regarding the user certificate and the assertion (authorization token)

Exceptions	• N/A
Preconditions	• Already authenticated to eIDAS • Already register and acquire Certificate from the CA • Already mapped with the certificate with the anonymous assertion

3.10.2 PP Security APIs

Operation requestPseudonym

Name	**requestPseudonym**
Description	This functionality returns the pseudonym to the S-EHR App.
Arguments	• access_condidtion token: String that includes an encrypted and signed message of the access condition (successful or not) of the mapping.
Return Value	• public key: String that includes the signed public part of the pseudonym for verification purposes • secret key: String that includes the signed pseudonym
Exceptions	• N/A
Preconditions	• Already authenticated to eIDAS • Already register and acquire Certificate from the CA • Already mapped with the certificate with the anonymous assertion

Operation publicKeyOwner

Name	**publicKeyOwner**
Description	This functionality is used for resolution of the identity of the patient for emergency purposes.
Arguments	• public key: String that includes the signed public part of the pseudonym for verification purposes
Return Value	• access_condition token: String that includes an encrypted and signed message of the access condition (successful or not) of the mapping.
Exceptions	• N/A
Preconditions	• Already authenticated to eIDAS • Already register and acquire Certificate from the CA • Already mapped with the certificate with the anonymous assertion

4 CONCLUSIONS AND NEXT STEPS

In this report, we defined the second and final version of the specification of remote and D2D identity management including authentication mechanisms for HRs interoperability. A technical background with state-of-the-art protocols and standards is also provided. More specifically, this deliverable includes the detailed crypto models and identity management 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). IDM and authentication in the D2D protocol will support two variants: a) identification with the ID-Card of the citizen and a QR code generated by the hospital including digital signatures from the HCP and the HO and b) identification with hardware-based digital signatures (e.g. qualified digital signatures) from both sides. In the R2D protocol we utilize the architecture of eIDAS to acquire the medical data for the first time from eIDAS-enabled EMRs. In addition the same eIDAS assertion will be utilised in the context of RDS for authentication purposes on the Pseudonym Provider (with anonymous assertion) in the second variant. Last but not least the authentication procedures in the R2D Backup and R2D Emergency are also included based on a simple username/password and ABAC-based mechanisms respectively. This final version of the deliverable acts as the detailed specification of identity management and authentication purposes defined in the context of InteropEHRate.

REFERENCES

- **[JOSANG2005]** A. Jøsang and S. Pope, ‘‘User centric identity management,’’ in Proc. *AusCERT Asia Pacific Inf. Technol. Security Conference*, p. 77, 2005.
- **[CARRETERO2018]** J. Carretero, G. I.-Moreno, M. V. Cabezas and J. G. Blas, “Federated Identity Architecture of the European eID System”, Digital Object Identifier, 2018
- **[EMTG2017]** E. M. Torroglosa-García and A. F. Skarmeta-Gómez, ‘‘Towards interoperability in identity federation systems,’’ J. Wireless Mobile Netw., Ubiquitous Comput., Dependable Appl., vol. 8, no. 2, pp. 1–25, 2017.
- **[SCUDDER2010]** Scudder, J., and Josang, A. “Personal federation control with the identity dashboard”. In IDMAN, Springer, pp. 85–99, 2010.
- **[IT2011]** Information Technology—Security Techniques—”A Framework for Identity Management”, Standard ISO/IEC 24760-1, Dec. 2011
- **[EU2009]** Digital Single Market, “Person Identification and Authentication – Key to eHealth and eGovernment Service”, 2019.
- **[FITZPATRICK2005]** B. Fitzpatrick, “Distributed Identity: Yadis,” May 2005.
- **[OPENID2011]** A. S. G. S. Gilbertson, “OpenID: The Webs Most Successful Failure,” Jan. 2011.
- **[SAML2005]** OASIS Security Services, “SAML Specifications | SAML XML.org,” Mar. 2005.
- **[XACML]** OASIS, XACML Oasis Specification. Website: https://www.oasisopen.org/committees/tc_home.php?wg_abbrev=xacml
- **[OAUTH2010]** E. Hammer-Lahav, “The OAuth 1.0 Protocol,” Apr. 2010. Website: https://tools.ietf.org/html/rfc5849
- **[STORK2010]** Vasilis Koulolias, “Secure idenTity acrOss boRders linKed (STORK)”, 2010, Website: https://joinup.ec.europa.eu/document/secure-identity-across-borders-linked-stork
- **[EPSOS]** epSOS, “epSOS Technical Aspects”, Website: http://www.promisalute.it/upload/mattone/gestionedocumentale/epSOS_Technical_Aspects_784_2462.pdf
- **[IHE ITI TF-1]** IHE IT Infrastructure (ITI) Technical Framework, “Integrating the Healthcare Enterprise”, Volume 1: Integration Profiles. Revision 11.0 – Final Text, September 23, 2014. Website: http://www.ihe.net/uploadedFiles/Documents/ITI/IHE_ITI_TF_Vol1.pdf
- **[ALTICELABS2014]** ALTICE LABS WHITEPAPER, “Identity and Access Management”, 2014. Website: https://www.alticelabs.com/content/WP-Information-Access-Control-Models.pdf
- **[ESENS2017]** Masi , M., Bittins, S. Cunha, J. and Atzeni, A., “e-SENS 5.2 eHealth eIDAS eID Pilot: Technical Feasibility Report”, 2017.
- **[EE2017]** European Commission, “Trust Services and eID (eIDAS regulation),” 2017, Website: https://ec.europa.eu/digital-single-market/en/trust-services-and-eid
- **[PKCS11]** OASIS, OASIS PKCS 11 TC, 2018 Website: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=pkcs11
- **[KAI2009]** R. Kai, Denis Royer and André Deuker, “The future of identity in the information society: Challenges and opportunities”, Springer Science & Business Media, 2009.
- **[NISTDSS]** FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, “Digital Signature Standard (DSS)”, 2013, Website: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf
- **[NGUYEN2018]** Nguyen, K., “Certification of eIDAS trust services and new global transparency trends”, Datenschutz und Datensicherheit, 2018

- **[KENNEDY2016]** Kennedy, E. and Millard, C., "Data security and multi-factor authentication: Analysis of requirements under EU law and in selected EU Member States", Computer Law & Security Review, 32/1, 91-110, 2016.
- **[KATEHAKIS2017]** Katehakis, D.G, Gonçalves, J., Masi, M., and Bittins, S., "Interoperability Infrastructure Services to Enable Operational Secure Cross-Border eHealth Services in Europe", 17th International HL7 Interoperability Conference IHIC, 2017.
- **[UAF2017]** FIDO Alliance, "FIDO UAF Architectural Overview", 2017 Website: https://fidoalliance.org/specs/fido-uaf-v1.1-ps-20170202/FIDO-UAF-COMPLETE-v1.1-ps-20170202.pdf
- **[U2F2017]** FIDO Alliance, "Universal 2nd Factor (U2F) Overview", 2017 Website: https://fidoalliance.org/specs/fido-u2f-v1.2-ps-20170411/fido-u2f-overview-v1.2-ps-20170411.pdf
- **[SAML2]** OASIS, "Security Assertion Markup Language (SAML) V2.0 Technical Overview", Committee Draft 02, 2008. Website: https://www.oasis-open.org/committees/download.php/27819/sstc-saml-tech-overview-2.0-cd-02.pdf
- **[eIDAS2018]** eIDAS-Node National IdP and SP Integration Guide, Version 2.1, 2018.
- **[RC6749]** FInternet Engineering Task Force, "The OAuth 2.0 Authorization Framework", 2012. Website: https://tools.ietf.org/html/rfc6749
- **[MOBILECONNECT]** GSMA, "Mobile Connect for Cross-Border Digital Services Lessons Learned from the eIDAS Pilot", 2019. Website: https://mobileconnect.io/wp-content/uploads/2019/02/MC-for-cross-border-digital-services_eIDAS_Feb2018-FINAL-web-2.pdf
- **[WSFEDERATION]** OASIS, "Web Services Federation Language (WS-Federation) Version 1.2", 2009. Website: http://docs.oasis-open.org/wsfed/federation/v1.2/ws-federation.html
- **[BERBECARU2019]** Diana Berbecaru, Antonio Lioy and Cesare Cameroni, "Electronic Identification for Universities: Building Cross-Border Services Based on the eIDAS Infrastructure", *MDPI Information*, 2019.
- **[RCF6749]** Internet Engineering Task Force, "The OAuth 2.0 Authorization Framework", Request for Comments: 6749, 2012.
- **[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
- **[D2.6]** InteropEHRate Consortium, *D2.6 - InteropEHRate Architecture - V3*, 2021. https://www.interopehrate.eu/resources/#dels
- **[D4.3]** InteropEHRate Consortium, *D4.3 - Specification of remote and D2D protocol and APIs for HR exchange V3*, 2021. https://www.interopehrate.eu/resources/#dels
- **[D3.3]** InteropEHRate Consortium, *D3.3 - Specification of remote and D2D IDM mechanisms for HRs Interoperability - V1*, 2019. https://www.interopehrate.eu/resources/#dels
- **[D3.6]** InteropEHRate Consortium, *D3.6 - Specification of data encryption mechanisms for mobile and web applications - V2*, 2021. https://www.interopehrate.eu/resources/#dels
- **[D3.8]** InteropEHRate Consortium, *D3.8 - Specification of consent management and decentralized authorization mechanisms for HR Exchange - V2*, 2021. https://www.interopehrate.eu/resources/#dels

- **[D4.8]** InteropEHRate Consortium, *D4.8 - Specification of protocol and APIs for research health data sharing - V1*, 2021. https://www.interopehrate.eu/resources/#dels
- **[QUALIFIED DEVICES]** European Commission, "Compilation of Member States notification on SSCDs and QSCDs", 2019. Website: https://ec.europa.eu/futurium/en/content/compilation-member-states-notification-sscds-and-qscds
- **[Gisdakis 2013]** S. Gisdakis, M. Laganà, T. Giannetsos and P. Papadimitratos, "SEROSA: SERvice oriented security architecture for Vehicular Communications," *2013 IEEE Vehicular Networking Conference*, Boston, MA, USA, 2013, pp. 111-118, doi: 10.1109/VNC.2013.6737597.
- **[eHDSI2021]** eHDSI Business Analyst, Ensure Health Professional (HP) Identification, Authentication and Authorization, 2021, https://ec.europa.eu/cefdigital/wiki/display/EHOPERATIONS/01.+Ensure+Health+Professional+%28HP%29+Identification%2C+Authentication+and+Authorization
- **[LABIOD2018]** Intercor project, Milestone 5 - Common set of upgraded specifications for PKI and Common Certificate Policy (CP), 2018, https://intercor-project.eu/wp-content/uploads/sites/15/2019/03/InterCor_M5_Upgraded-Specifications-PKI-CP_v1.0_INEA-1.pdf

6 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
σ	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 1 - 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 helloSEHR message


```
{
        "header": {
                "timeStamp": "2021-07-26T14:13:13.553Z",
                "agent": "JRE 1.8.0_261 - Windows 10 10.0",
                "protocol": "D2D",
                "version": "1"
        },
        "operation": "HELLO_SEHR",
        "body": "{\"resourceType\":\"Practitioner\",\"id\":\"2ef69593-1408-411b-a089-
08dcfc97d1f4\"...}"
}
```


JSON sample for helloHCPApp message


```
{
        "header": {
                "timeStamp": "2021-07-26T14:13:14.553Z",
                "agent": "JRE 1.8.0_261 - Windows 10 10.0",
                "protocol": "D2D",
                "version": "1"
        },
        "operation": "HELLO_HCP",
        "body": "{\"resourceType\":\"Organization\",\"id\":\"2ef12345-1408-216y-a089-
08dcfc97d1f4\"...}"
}
```