CrowdHEALTH

Collective Wisdom Driving Public Health Policies

Del. no. – D4.20 Integrated Holistic Security and Privacy Framework: Software Prototype v1

Project Deliverable

This project has received funding from the European Union's Horizon 2020 Programme (H2020-SC1-2016-CNECT) under Grant Agreement No. 727560

D4.20 Integrated Holistic Security and Privacy Framework: Software Prototype v1

Work Package:	WP4
Due Date:	31/12/2017
Submission Date:	18/01/2018
Start Date of Project:	01/03/2017
Duration of Project:	36 Months
Partner Responsible of Deliverable:	UPRC
Version:	1.0
Status:	☒ Final ☐ Draft ☐ Ready for internal Review ☐ Task Leader Accepted ☐ WP leader accepted ☒ Project Coordinator accepted
Author name(s):	Stefanos Malliaros (UPRC), George Moldovan (SIEMENS), Christos Xenakis (UPRC)
Reviewer(s):	Antonio De Nigro (ENG), Dimitris Miltiadou (SiLO)
Nature:	☐ R – Report ☒ D – Demonstrator
Dissemination level:	☒ PU – Public ☐ CO – Confidential ☐ RE – Restricted

REVISION HISTORY			
Version	Date	Author(s)	Changes made
0.1	22/11/2017	UPRC	TOC to share with the participants.
0.2	06/12/2017	UPRC	Revised TOC and added content
0.3	07/12/2017	UPRC	Added Content
0.4	08/12/217	UPRC	Added Content
0.5	11/12/2017	UPRC	Added Content
0.6	12/12/2017	UPRC	Added Content
0.7	13/12/2017	UPRC	Added Content
0.8	18/12/2017	SIEMENS	Added Content
0.9	16/01/2018	UPRC	Addressed Comments from Peer Review
0.9.1	17/01/2018	UPRC	Addressed Comments from Peer Review
1.0	18/01/2018	ATOS	Quality review. Submission to EC.

List of acronyms

ABAC	Attribute Based Access Control
API	Application Programming Interface
CLIPS	C Language Integrated Production System
CSV	Comma-separated values
DNS	Domain Name System
ECA	Event Condition Acction
FIDO	Fast IDentity Online
FQDN	Fully qualified domain name
GPG	Gnu Privacy Guard
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
JDBC	Java DataBase Connectivity
LDAP	Lightweight Directory Access Protocol
OAuth	Open Authentication
OS	Operating System
SAML	Security Assertion Markup Language
SCIM	System for Cross-domain Identity Management
SSO	Single Sign On
UMA	User-Managed Access
URI	Uniform Resource Identifier
URL	Uniform Resource Locator

Contents

1. Executive Summary 6
2. Introduction 7
3. Prototype overview 8
3.1. Main components of the prototype 8
3.1.1. User authentication, authorization, and access control 8
3.1.2. Data anonymization 9
3.1.3. Trust and Reputation Modelling 10
3.2. Interfaces 12
3.2.1. User authentication, authorization, and access control 12
3.2.1.1. oxTrust Admin UI 12
3.2.1.2. oxAuth OP 13
3.2.2. Data anonymization 16
3.2.2.1. ARX: Configuration phase 16
3.2.2.2. ARX: Exploration phase 18
3.2.3. Trust Management and reputation modelling ally 18
3.3. Baseline technologies and tools 20
3.3.1. CLIPS rules 20
3.3.2. Event Condition Action support 20
4. Source code 21
4.1. Availability 21
4.2. Exploitation 21
4.2.1. OpenID Connect Provider 21
4.2.2. Data anonymization 24
4.2.3. Trust and reputation modelling 24
5. Conclusions 26
References 27

List of figures

Figure 1: User authentication, authorization, and access control 8
Figure 2: Data anonymization Process 10
Figure 3 Trust and reputation components 11
Figure 3: Organization Configuration in oxTrust admin UI 12
Figure 4: Manage user registration in oxTrust Admin UI 13
Figure 5: Authorization Code Flow in OpenID Connect 15
Figure 6: Implicit Code Flow in OpenID Connect 16
Figure 7: ARX Configuration phase user interface 17
Figure 8: ARX Exploration phase user interface 18
Figure 9: Gluu oxTrust Admin UI 24

1. Executive Summary

This document is a part of the WP4: Information and Knowledge Acquisition and Management of the CrowdHEALTH project. This deliverable aims to report and describe the first version of the software prototype of the Integrated Holistic Security and Privacy Framework. The first version of the design and specification phase of the Security and Privacy Framework was presented in D4.17.

The purpose of this deliverable is to present and describe the implementation of the components that constitute the Security and Privacy Framework. These components aim in providing user authentication, authorization and access control, together with trust management and reputation modelling. The first part of this document is to overview the different components of the Security and Privacy Framework by specifying their architecture. Next, the interfaces that will be used by the users of CrowdHEALTH are presented. Each component of the Security and Privacy Framework is presented independently, and any interconnection between them is analysed accordingly, together with the baseline technologies that is utilizes. Lastly, this document provides information on how to install and deploy the components of the CrowdHEALTH's Holistic Security and Privacy Framework, by providing the necessary code snippets.

2. Introduction

This aim of the CrowdHEALTH project is to design and implement an architecture which will process, analyse, and extract knowledge from the analysis of anonymized electronic health records. The stakeholders of this project can be citizens, policy makers, governmental organization, researchers, healthcare providers, and doctors. This deliverable describes the software prototype of the Holistic Security and Privacy Framework, as specified in D4.17: Integrated Holistic Security and Privacy Framework: Design and Open Specification v1 [1]. To elaborate, the components of the security framework provide: (i) trust management to quantify the trustworthiness of the CrowdHEALTH entities, (ii) data anonymization, and (iii) user authentication, authorization, and access control. This deliverable analyses the architecture, the user interfaces, the baseline technologies, the source code availability, and the exploitation potential for the components that offer each one of the above mentioned services. The next version of this deliverable will be D4.21: Integrated Holistic Security and Privacy Framework: Software Prototype v2, and will extend the implementation status of the security and privacy framework components.

The structure of this document is as follows: Section 3 is composed of three subsections and presents each component of the CrowdHEALTH Integrated Holistic Security and Privacy Framework in detail. In section 3.1 the architecture of each component is presented, while section 3.2 elaborates on the usability of each component by presenting its user interfaces. The baseline technologies that are employed by the components of the Holistic Security and Privacy Framework are presented in section 3.3. Section 4 elaborates on the availability and the exploitation of the components presented in section 3. Particularly, section 4.1 states the availability of the source code, while section 4.2. present detailed instructions for installing and exploiting each component. Finally, section 5 concludes by summarizing the status of the CrowdHEALTH's Integrated Holistic Security and Privacy Framework.

3. Prototype overview

3.1. Main components of the prototype

The initial design and specification of the Holistic Security and Privacy Framework is described in D4.17 [1]. In the following subsections, each component will be presented separately, and its implementation status will be denoted.

3.1.1. User authentication, authorization, and access control

The CrowdHEALTH's user authentication and authorization component uses the open source protocol OpenID Connect [2]. The OpenID Connect introduces an authentication layer, on top of OAuth 2.0 [3], which is an authorization framework, that enables applications to obtain limited access to user accounts on an HTTP service. This is performed by delegating user authentication to the service that hosts the user account, and authorizing third-party applications to access the user account. The CrowdHEALTH applications include an Attribute Based Access Control (ABAC) module, which is responsible for enforcing access control policies, based on the attributes of the Open ID Connect Provider userbase. The main components of the user authentication, authorization, and access control mechanism of CrowdHEALTH are shown in Figure 1.

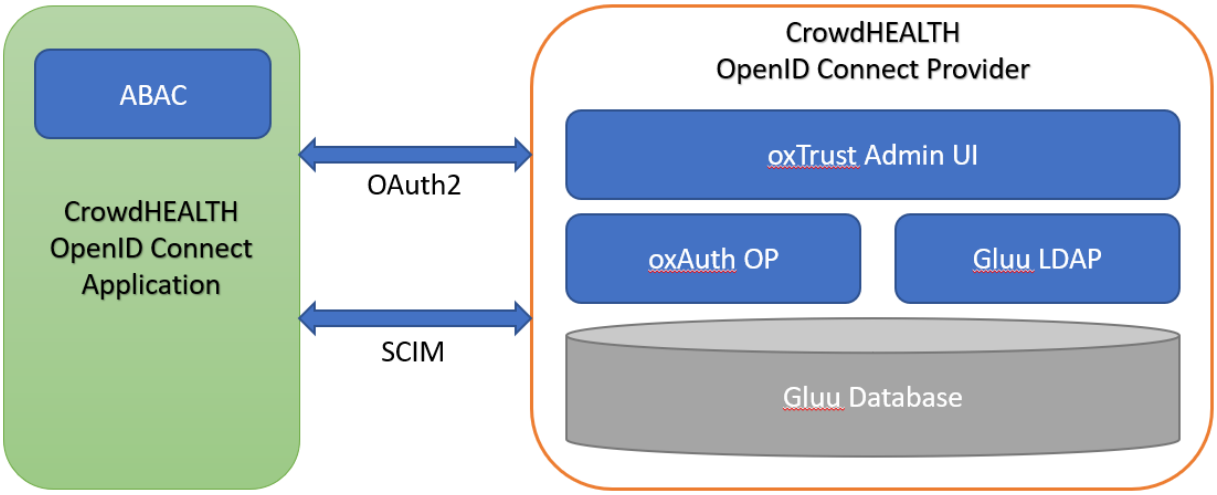

Figure 1: User authentication, authorization, and access control

The CrowdHEALTH OpenID Connect Provider is the main component of the user authentication, authorization, and access control module. The first layer of the OpenID Connect Provider is its internal database, named Gluu Database. The database contains the users' attributes, emails, usernames, and passwords. Moreover, it also stores information about the applications, that are authorized to request identity information, such as a secret value shared between the OpenID Connect Provider and the application, the application ID, the application type, and the login and redirect URIs. There is also specific information

regarding the authentication method used for the relevant application, the scopes it can access and the types of tokens it can request. The second layer of the OpenID Connect Provider is composed of the “oxAuth OP”, and “Gluu LDAP” modules. These modules access and modify the contents of the Gluu database related to their functionality. The oxAuth OP module is the one that runs the OpenID Connect protocol, and the authorization server. Applications that would like to access user attributes interact with the oxAuth OP module by using the OAuth 2.0 connect flows, and including the OpenID Connect scopes. The second module of the middle layer of the OpenID Connect Provider is Gluu LDAP. This module is responsible for maintaining the user information in the database, by using the Lightweight Directory Access Protocol (LDAP) [4] application protocol for accessing and maintaining distributed directory information services. The top layer of the OpenID Connect Provider is the oxTrust Admin User Interface, which is the administration application, and can be accessed via a web browser. By using this module, the administrator can configure all the components of the OpenID Connect Provider, manage the user database, register or delete applications, and define user attributes that can be shared to authenticated applications.

The CrowHEALTH’s OpenID Connect Provider is based on Gluu server [5], which is an open source software under the MIT license [6]. We selected Gluu as the basis of CrowdHEALTH’s implementation for several reasons. The first and most important is that Gluu is certified with the OpenID Certification [7] and conforms with all the different profiles defined by the OpenID Connect standard. The Gluu server uses open web standards for authentication, authorization, federated identity and identity management such as OpenID Connect [2], OAuth 2.0 [3], SAML 2.0 [8], User Managed Access 2.0 (UMA) [9], System for Cross-domain Identity Management (SCIM) [10], FIDO Universal 2ND Factor (U2F) [11], and Lightweight Directory Access Protocol (LDAP) [4]. Gluu also offers a wide variety of use cases, and the most common of them are: Single Sign-On (SSO) [12], Mobile authentication, API access management, Two-factor authentication, Customer identity and access management, and Identity Federation.

3.1.2. Data anonymization

The data anonymization is one of the most significant processes in CrowdHEALTH, since the data that will be processed contain sensitive information. Due to the severity of the anonymization process, CrowdHEALTH will employ the use of ARX [13], which is a successful, and open source data anonymization tool. It is a software for privacy-preserving microdata publishing, as it implements methods of statistical disclosure, and supports several privacy models. These models mainly use the generalization and suppression methodologies to compute an optimal solution with minimal loss of data quality [14]. The success of this tool can be justified, due to the numerous studies that have been performed to anonymized datasets [15]. The main processes of the ARX anonymization tool, are shown in the figure below:

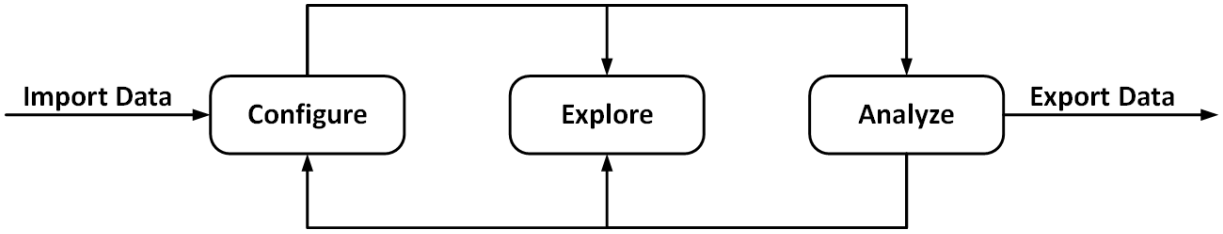

Figure 2: Data anonymization Process

The Data anonymization process that will be followed in CrowdHEALTH consists of the following phases:

- **Configuration:** In this phase, the data are imported and all the necessary anonymization parameters are configured. The data can be imported from various sources with user defined syntax. One of the most significant steps of this phase is the creation of the generalization hierarchies, which are used to anonymize quasi-identifiers. The hierarchies presented in intervals, and can be imported via CSV files.
- **Exploration:** A search space is created by using a set of transformations to the applied datasets. By using the exploration phase, the users are assisted to choose the appropriate anonymization methodology.
- **Analysis:** The last phase of the anonymization process, which consists of two parts, is the analysis of the created data. The analysis is performed initially from the perspective of data utility. From this perspective, the user finds the suitability of the selected transformation for the given usage scenario. This is performed by depicting statistical measurements from both anonymized and de-anonymized data side by side. The second perspective is the risk analysis, in which metrics are used to estimate and present the privacy risk.

3.1.3. Trust and Reputation Modelling

The trust and reputation modelling component within CrowdHEALTH is composed of three distinct sub-components:

- A **trust evaluation** component: which reflects the individual trust rating for a specific data object, provided service or whole sensor from the perspective of a single observer. Multiple observers can function by employing different trust models, thus generating different trust evaluations for the same entity (here forth used interchangeable with the previous set of possible interactions, at data, service or sensor levels).
- A **reputation evaluation** component: which merges and generates a single reputation rating for a specific entity, based on the inputs from one or multiple observers. At any time, there is a single reputation rating available, unlike the trust evaluation which can be very subject. In addition, the reputation evaluation reflects the rating of a specific

entity by the system, and is being used by components within the system which have not interacted directly with it.

- A **reactive** component: which, on certain reputation thresholds, can generate certain events. Usually we refer to alarms, specific tags used within the system, and similar. Reactions are also meant to drive and adjust the interactions of the CrowdHEALTH system with specific entities. E.g. by refusing communication with those which have negative reputation evaluations in certain contexts, such as when a certain value cannot be verified by corroboration.

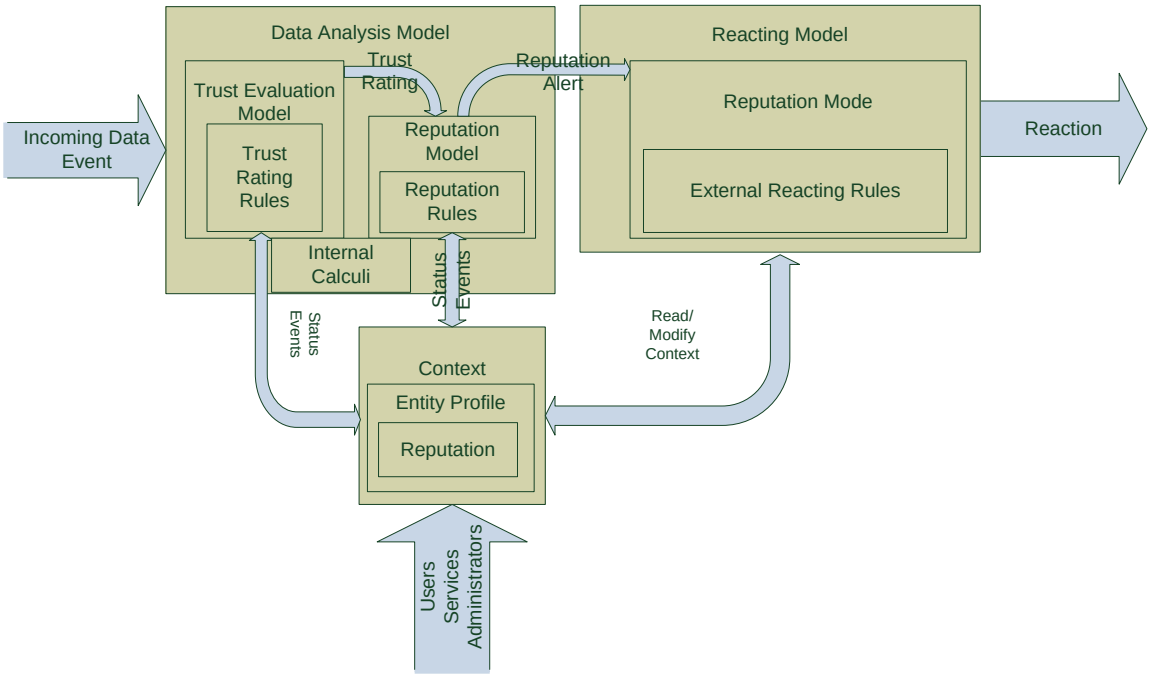

Figure 3 Trust and reputation components

The interactions between the tree components is depicted in Figure 3. In addition, a context-specific information is being feed into the system, which assists and provides additional information relevant to the various functioning reputation and trust models.

Technical wise, the current prototype for the trust and reputation component is written as Spring-based Java microservices. Note-worthy is the current employment of the CLIPS (C Language Integrated Production System) [16] engine for performing expert system-related operations. In the case of the trust and reputation component, the CLIPS rules are used to drive the observer interactions with the observed entity by computing trust ratings.

3.2. Interfaces

3.2.1. User authentication, authorization, and access control

3.2.1.1. oxTrust Admin UI

The oxTrust admin UI can be reached via a web browser by pointing to the IP address or the FQDN of the Gluu OpenID Connect Provider. After logging in with the username and password, the welcome page presents basic information, such as the Gluu server version, the free memory, the system uptime, and the free disk space. The configuration and the navigation to the various functionalities and configurations is performed via the left-hand menu.

In the configuration tab of the oxTrust admin UI, the administrator can modify and set non-protocol related features. Particularly, the administrator can enable the self-service password reset, the SCIM support, set the DNS servers, allow the users to edit their own profile, and set a contact email address for the server's administrator. Moreover, the administrator can specify the mail server properties, that will be used to send emails, and specify whitelist or blacklist IP addresses, to secure the oxTrust Admin UI webpage. Another significant functionality, that can be managed through the oxTrust admin UI, is the user registration. The user registration is performed via two different approaches. The first one is that the user registration will be performed via the applications by using the SCIM protocol.

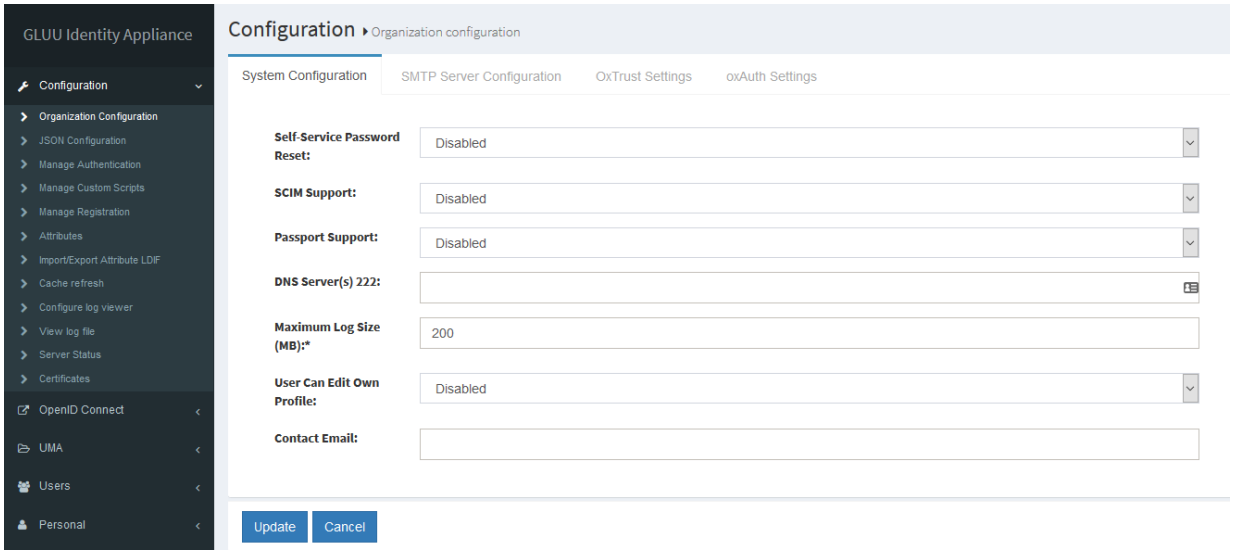

Figure 4: Organization Configuration in oxTrust admin UI

The second, is the self-enrollment of the users that can be performed through the URL: *http://<hostname>/identity/register*. To enable this functionality, we need to click the *Enabled* textboxes, in the Manage *Custom Scripts* → *User Registration* tab. Although the default registration form, contains predefined attributes, the administrator can specify new attributes,

that are required through registration, by navigating to the *Organization Configuration → Manage Registration*. Lastly, Gluu server by default supports manual approval of user registration. If an administrator wants to enable user login, as soon as they complete their registration, we need to modify the *enable_user* parameter in the *Manage Custom Scripts → User Registration* tab to true.

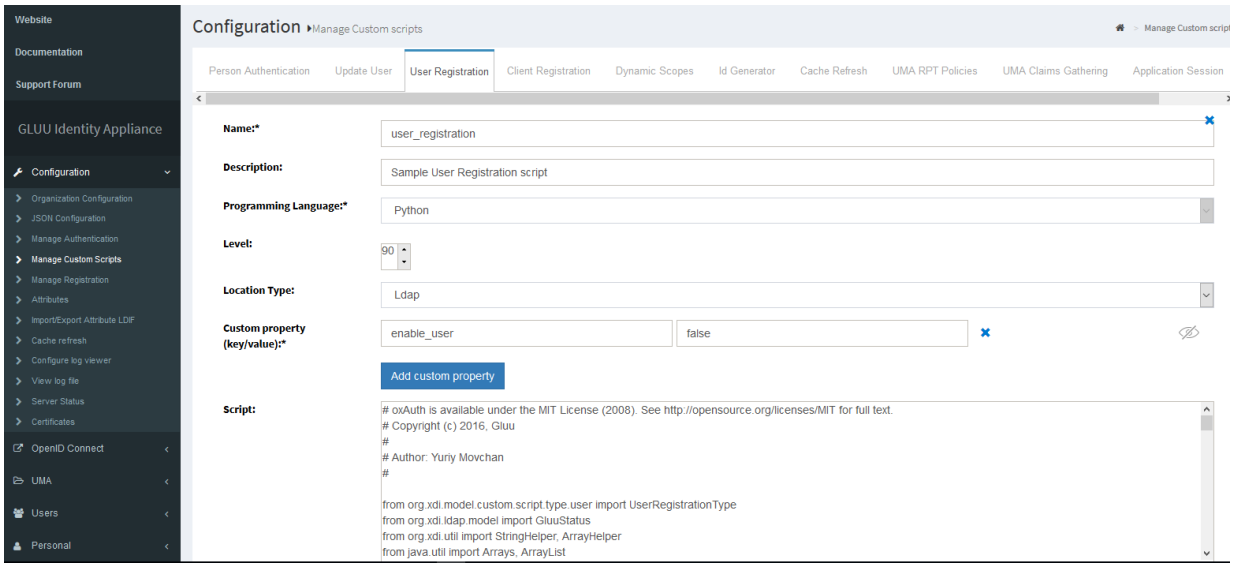

Figure 5: Manage user registration in oxTrust Admin UI

3.2.1.2. oxAuth OP

oxAuth OP is the module that runs the OpenID Connect provider protocol stack, and it has passed all the OpenID Provider conformance profiles [7]. Particularly, it supports the following OpenID specifications:

- **Core:** It is the main specification of Open ID Connect. It enables applications to verify the identity of the end user based on the authentication performed by an authorization server, as well as obtain basic profile information about the end user in an interoperable an REST-like manager.
- **Dynamic Client Registration:** Each application that has can authenticate users, based on the Gluu userbase, is registered by the Gluu server administrator. By enabling the Dynamic Client Registration feature, applications can automatically register to the Gluu server, and authenticate their users based on the Gluu server userbase. To disable the Dynamic Client Registration can be disabled by setting the `dynamicRegistrationEnabled` value to `False` in the oxAuth JSON properties.
- **Discovery:** The OpenID Connect Provider exposes the available configuration files, so that automatic configuration of applications is feasible. The Gluu configuration endpoint can be reached by accessing the https://{hostname}/.well-known/openid-configuration web page via a web browser.

- **Form Post Response Mode:** The Form Post Response Mode is an optional specification in OpenID Connect. By using this method, authorization response parameters are encoded and passed as HTML form data, to the Application by an HTML POST request.
- **Session Management:** By using this protocol, we can find the login status of the end-user at the OpenID Connect Provider. This is essential, to discover if the end user might have logged the OpenID Connect Provider before the expiration time of the token.

The two main flows that the OpenID Connect protocol proposes are the Authorization code flow, and the Implicit flow. The authorization code flow is used by applications, that have a backend, which needs to access user information periodically. The steps that are followed for the authorization code flow are the followings:

- The user accesses the application.
- The application prepares an authentication request, containing the requesting parameters, and this request is sent to the OpenID Connect Provider. The `response_type requested` parameter in the authorization code flow is `code`.
- The end user is redirected to the login page of the OpenID Connect provider, in which the user is verified and authenticated.
- The OpenID Connect provider, redirects the user to the application with `authorization code`.
- The `authorization code` is sent to the OpenID Connect Provider by the application.
- The OpenID Connect Provider, verifies the `authorization code`, and creates an `identity token` and an `access token`. These tokens are sent to the application.
- The application sends the `identity token` to the `token endpoint` of the OpenID Connect Provider to get access to the end user's information. Next, the application can access the `UserInfo endpoint`, by showing the `access token`, and request the required claims of the end user.

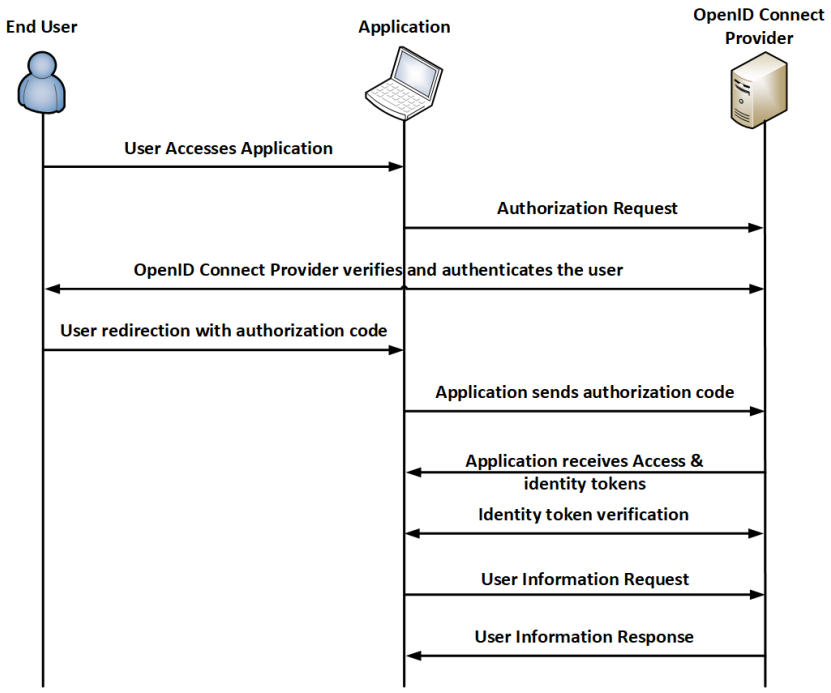

Figure 6: Authorization Code Flow in OpenID Connect

The Implicit flow is used in Single Page Applications (SPAs), or in clients implemented into web browsers, since the applications do not request periodically user information from the OpenID Connect Provider. The steps of the implicit flow are the following:

- The user accesses the application.
- The application prepares an authentication request, containing the requesting parameters, and this request is sent to the OpenID Connect Provider. The `reponse_type requested` parameter in the authorization code flow is `id_token` or "`id_token token`".
- The end user is redirected to the login page of the OpenID Connect provider, in which the user is verified and authenticated.
- The OpenID Connect provider redirects the user to the application with an `id token` and an `access token`.
- The application uses the `id token` to authorize the end user.
- The application can access the `UserInfo endpoint` for getting the requested user claims.

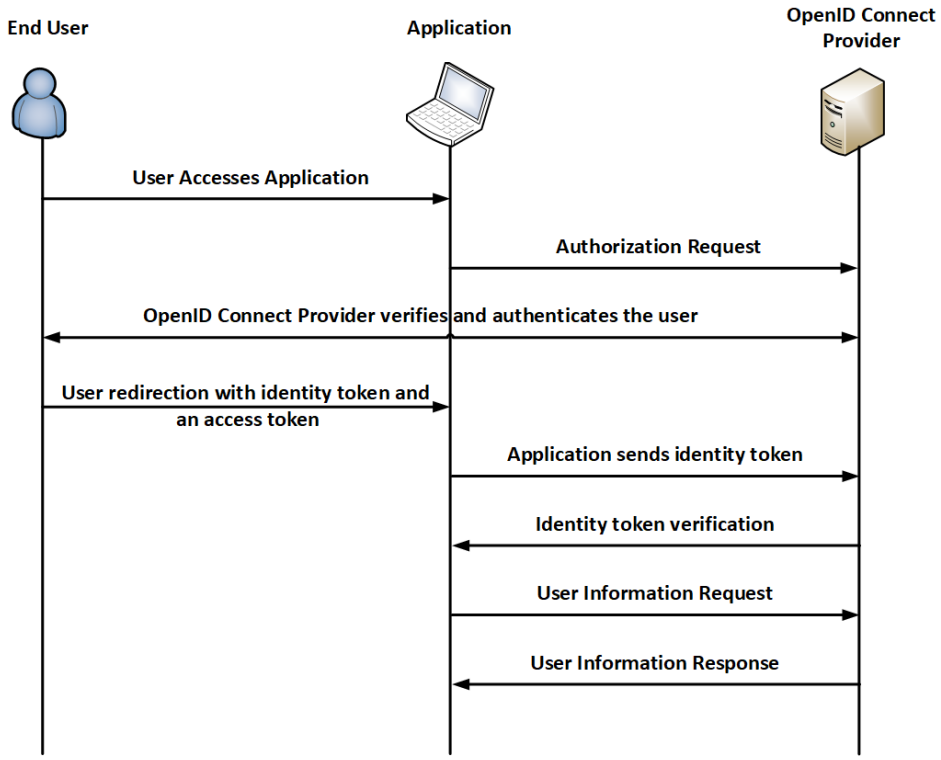

Figure 7: Implicit Code Flow in OpenID Connect

3.2.2. Data anonymization

The ARX data anonymization tool can either be used via the provided GUI interface, or via the use of a Java programming API. In this version of the deliverable, we will cover the GUI of the ARX software.

3.2.2.1. ARX: Configuration phase

After opening ARX, the user views the configuration tab, which is comprised of five areas, as shown in the figure below. Area 1 is the current data set that has been imported into ARX. The import process, can be performed by using different input methods, such as CSV files, Excel Files, and JDBC databases. In Area 2, the user can define the type of each attribute, and define generalization rules in intervals, ordinal and mask form for quasi-identifier and sensitive attributes. Area 3 is responsible for configuring the privacy model that will be used in that given scenario. CrowdHEALTH will employ the use of k-Anonymity [17], which aims at protecting datasets from identity disclosure. A detailed analysis of k-Anonymity can be found in D4.17 [1]. By using area 4, a user can configure the utility metrics, that will be used to evaluate the suitability of the selected transformation for the given scenario. Lastly, with area 5 a user can specify methods to anonymize and extract a subset of the overall data of that given scenario.

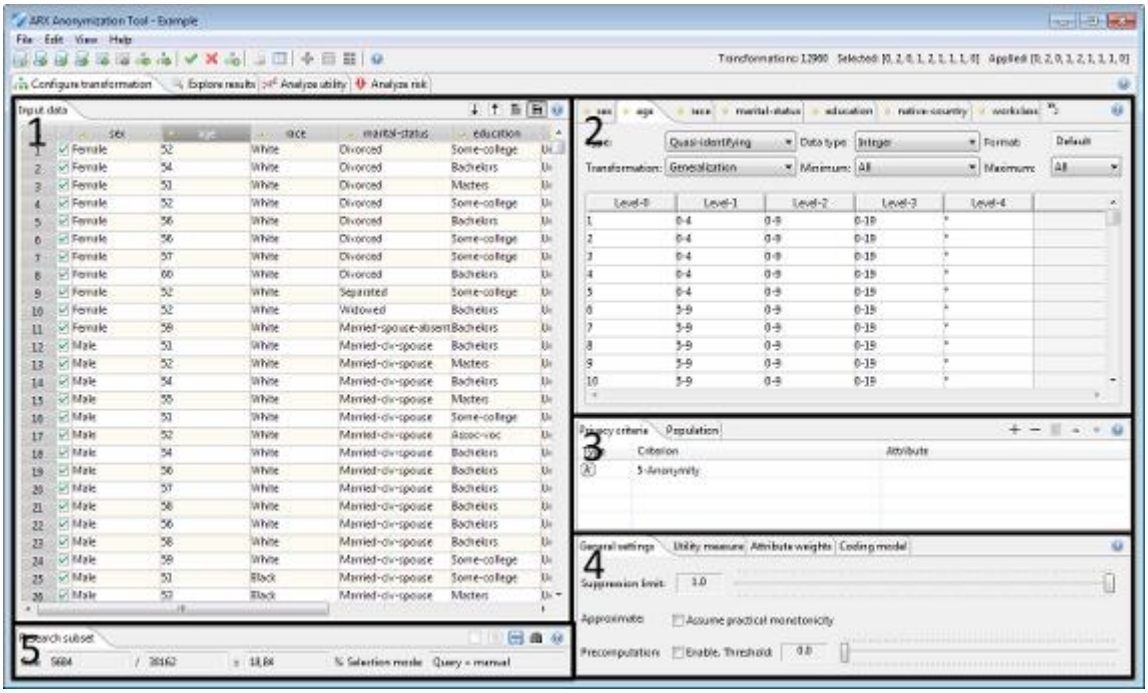

Figure 8: ARX Configuration phase user interface

The imported data set is shown as a table in area 1. In the header of each column, the user can indicate the type of each attribute, by selecting one of the following colors:

- Red: It is used for direct identifiers, which are associated with a substantial risk of reidentification. These values are removed from the anonymized data set.
- Yellow: This color indicates quasi-identifiers. These attributes are associated with a high risk of reidentification, but they will be anonymized via the generalization or suppression techniques.
- Purple: Purple indicates sensitive attributes, which represent information that the subject does not want to be linked with, such as diagnoses.
- Green: Insensitive attributes are indicated via the green colour. The values of the insensitive attributes will be kept unmodified.

The generalization hierarchies, that will be followed in the given scenario, are shown at the data transformation tab. The metadata tab can be used to specify the type (i.e. String, Integer, Decimal, Date, Ordered String) of attributes. Generalization hierarchies can be defined depending on the attribute types, and can be classified into three categories. Masking-based hierarchies are the most commonly used for a broad spectrum of attributes. Interval-based hierarchies can be used for attributes that are presented in ratio scale. Lastly, Order-based hierarchies can be applied into ordinal attributes.

3.2.2.2. ARX: Exploration phase

The exploration tab is used during the exploration phase, and its user interface is shown in the following figure. In area 1, a subset of the available solutions is presented, while area 2 applied filters for limiting the solution space in area 1. Area 3 contains all the detail of the transformations, while area 4 shows the properties of the transformation selected in area 3. Each node presented in area 1 states a solution, based on the generalization levels for the quasi-identifiers in the input data set. A transformation of Red colour shows that the underlying transformation does not result in a privacy preserving data set. Solutions of green colour result in privacy-preserving dataset, while the bets transformation, in terms of utility, is indicated with Yellow colour.

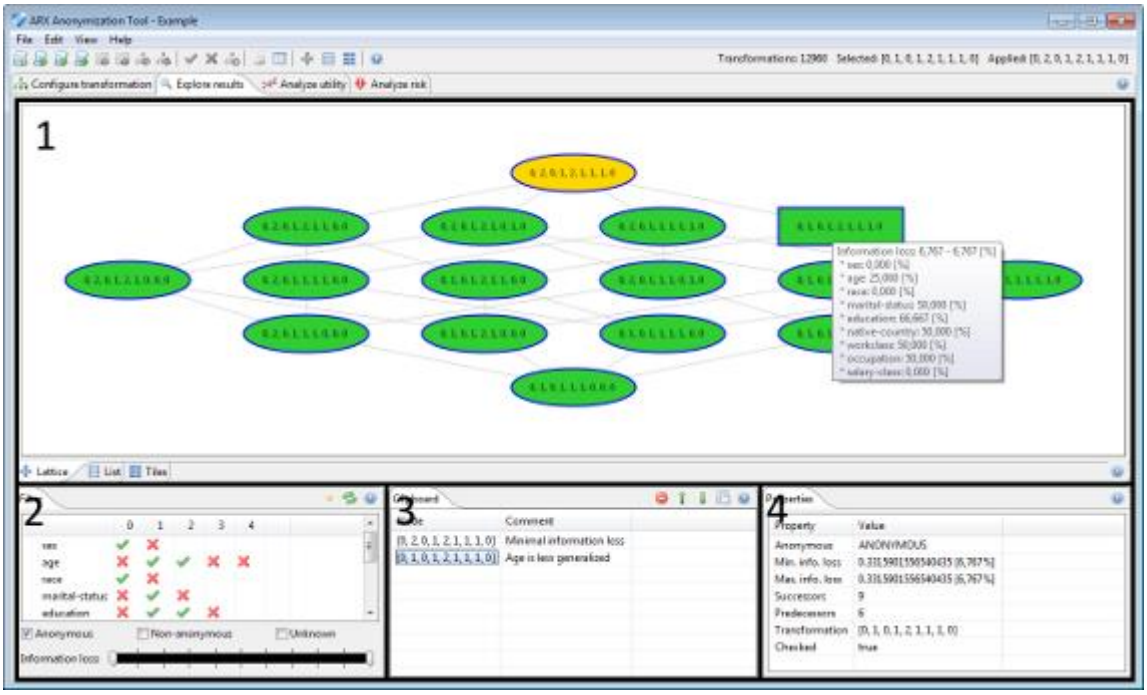

Figure 9: ARX Exploration phase user interface

3.2.3. Trust Management and reputation modelling ally

The modeling of trust and reputation engine makes use of written rules. In CLIPS, the rules have a format as detailed in Snippet 1.

```
;=======================

; Rule valC-mima

;=======================
(defrule valC-mima "checks valC (str val)a-priori boundary conditions
 of each observer [ 0 < valC < 40 ]"
      (a-valC-mima (obsN ?obsN)(strN ?strN) (ruID ?ruID)(minA
?minA)(maxA ?maxA))
      (a-str (strN ?strN)(valC ?valC) (timC ?timC))
      (test (or (< ?valC ?minA)  (> ?valC ?maxA)))
      =>
      (assert (bad ?strN ?obsN "valC-mima" ?ruID (getTime ?timC)))
      ;(printout t "Alert! "?obsN"'s "?strN" values are abnormal"
crlf)
)
```

Snippet 1 Example of CLIPS rule checking if a certain values is between a minimum and a maximum value.

For each statistics-based assertion of data, a rule has to be created. Relevant statistic operations are typical produced by algorithms such as the standard deviation (see Snippet 2), references to historical values, or the computing of reputation values.

```
;=================
; Rule aveE-stdE
;=================
(defrule aveE-stdE "checks if aveE [avgA-(zed)std < aveE <
avgA+(zed)std]"
      (a-aveE-stdE (obsN ?obsN)(strN ?strN) (ruID ?ruID) (zed ?zed)
(aveA ?aavg))
      (a-str (strN ?strN) (aveE ?aveE)(stdE ?estd) (timC ?timC))
      (test (or (< ?aveE (- ?aavg (* ?zed ?estd)))
                  (> ?aveE (+ ?aavg (* ?zed ?estd)))
              ))
      =>
      (assert (bad ?strN ?obsN "aveE-stdE" ?ruID (getTime ?timC)))
      ;(printout t "Alert! "?obsN" "?strN"'s aveE is abnormal
      ;compared to a-priori average (+-) "?zed" times the ewm-Standard
Deviaton" crlf)
)
```

Snippet 2 CLIPS rule of computing the standard deviation computatoin

3.3. Baseline technologies and tools

All the components of the Gluu server are written in Java, except the LDAP database and the apache server which are written in C. The APIs provided for the authentication, and authorization is written in Java, while the API provided for the user management, via the SCIM protocol is written in C. The ARX anonymization tool is also written in Java, but it provides installers for both Windows, Linux and MAC OS X operating systems.

3.3.1. CLIPS rules

CLIPS is an open source solution for expert systems, providing a complete object oriented language for writing expert system rules. Although it is written in C, there are wrappers for several other programming languages, such as Java.

3.3.2. Event Condition Action support

The reaction module within the trust and reputation component is meant to assist the system in executing proper, in-time actions when specific reputation thresholds are detected. More specifically, the reaction model is currently expected to execute following actions:

- Annotate log files,
- Generate warning or alert emails,
- Disable or enable services.

Implementation wise, the mechanisms makes use of the event condition action (ECA) framework, which denotes the structure of rules in the case of such event driven architectures:

- An event triggering the invocation of the rules,
- A condition, which executes a logical test,
- And an action, which consists of updates to the system.

Although not a pure ECA engine, Drools [18] offers a suitable solution for rule-based interactions and triggers, and is therefore adopted for the reaction-module prototype. Its advantages are the fact that it is easily integratable into Java applications, as well as the fact that it offers JSR-94 compliant mechanisms, which makes it compatible with legacy applications and Java tools.

4. Source code

4.1. Availability

The CrowdHEALTH's OpenID Connect Provider is based on Gluu server [5], which is an open source software and can be found available online [19].

The CrowdHEALTH's data anonymization component is based on ARX [13], which is an open source software and can be found available online [20].

4.2. Exploitation

4.2.1. OpenID Connect Provider

The Gluu OpenID Connect Provider can be installed on any physical or virtual machine, with at least 4 GB of RAM (8GB is recommended), 40 GB of Hard Drive space, and a 64bit processor. Gluu server should be deployed on systems with either Ubuntu 16.04, Centos 7.x, RHEL 7.x, or Debian 8. The following commands apply to Ubuntu Desktop 16.04.3, but they should be also relevant to other Linux distributions. After finishing the installation of Ubuntu 16.04.3, all the software should be updated to the latest version by running the following commands:

```
sudo apt-get update

sudo apt-get -y upgrade

sudo apt-get -y dist-upgrade
```

The next step after installing and updating the OS is to make some changes on how the file system handles the files. This is performed by adding the following two lines in the `/etc/security/limits.conf` file:

```
* soft nofile 65536

* hard nofile 262144
```

In order to preserve server resources ensure that the line `session required pam_limits.so` in the file `/etc/pam.d/login` is not commented. Also, ensure that the file descriptor limit value in the file `/proc/sys/fs/file-max` is at least 65535. Lastly, reboot the server for the changes to take effect.

After reboot, the Gluu server software can be installed by executing the following commands. These commands add the Gluu repository, and its' GPG Key to Ubunntu's software resources, and then updates the available software list, and installes the Gluu server software

```
sudo -s

apt-get install curl

echo "deb https://repo.gluu.org/ubuntu/ xenial main" >
/etc/apt/sources.list.d/gluu-repo.list

curl https://repo.gluu.org/ubuntu/gluu-apt.key | apt-key add -

apt-get update && apt-get -y install gluu-server-3.1.1
```

After reboot, we are ready to start, and configure the Gluu server. The Gluu server is designed to run in a chroot container, to avoid interaction with the rest server software. The Gluu server can start by executing the following commands. The first one starts the Gluu server, while the second changes the root directory so that the Gluu server can run.

```
sudo service gluu-server-3.1.1 start

sudo service gluu-server-3.1.1 login
```

After logging to the Gluu server environment, we need to complete the configuration script, which creates the certificates configuration files, and salt values. This can be achieved by executing the following commands in the chroot shell of Gluu. It is important to note that the configuration script can only run once, since another execution will break the Gluu sever instance.

```
cd /install/community-edition-setup

./setup.py
```

During the execution of the configuration script, the user is prompted to answer the following questions regarding the development environment:

- **Enter IP Address:** Provide the IP address of the server.
- **Enter hostname:** Provide the FQDN, or CNAME that the pc is assigned.
- **Enter your city or locality:** Provide City name for certificate generation.
- **Enter your state or province two letter code:** Provide state for certificate generation.
- **Enter two letter country code:** Provide two letter country code for certificate generation.
- **Enter organization name:** Provide organization name for certificate generation.
- **Enter email address for support at your organization:** Provide administrator's email address.
- **Enter maximum RAM for applications in MB:** Leave the default value.

- **Optional: enter password for oxTrust and LDAP superuser:** Copy the given password or enter your desired password. It is used as the LDAP directory manager password, and for the default admin user for oxTrust.
- **Install oxaAuth2 Authorization Server:** Press enter if you would like to install the embedded authorization server. This module is mandatory.
- **Install oxTrust Admin UI:** Press enter if you would like to install the embedded administration UI. This module is mandatory.
- **Install LDAP server:** LDAP server is used to store user info and configuration data. This module is mandatory.
- **Install Apache HTTPD server:** This is the web server that is used to access the oxTrust Admin UI. This module is mandatory.
- **Install Shibboleth SAML IDP:** This module if optional, and is necessary to install only if you need a SAML identity provider.
- **Install Asimba SAML Proxy:** This module is not recommended for installation, since the passport module has replaced it.
- **Install oxAuth RP:** oxAuth RP is an OpenID Connect test client. It is only recommended for test environments.
- **Install Passport:** Passport is the Gluu module for offering social login to the users.
- **Install JCE 1.8:** This module is mandatory, since I provide cryptographic extensions to the Gluu server.

After the installation has been completed, we can open the Gluu administration panel by pointing to the IP address of the FQDN. The default username is admin, and the password is the one that was entered during the configuration script. The oxTrust Admin UI is shown in the following figure. If we want to disable the Gluu repositories to avoid any updates that might break the server, we can comment out all the Gluu-related repositories in `/etc/apt/sources.list.d/gluu-repo.list`.

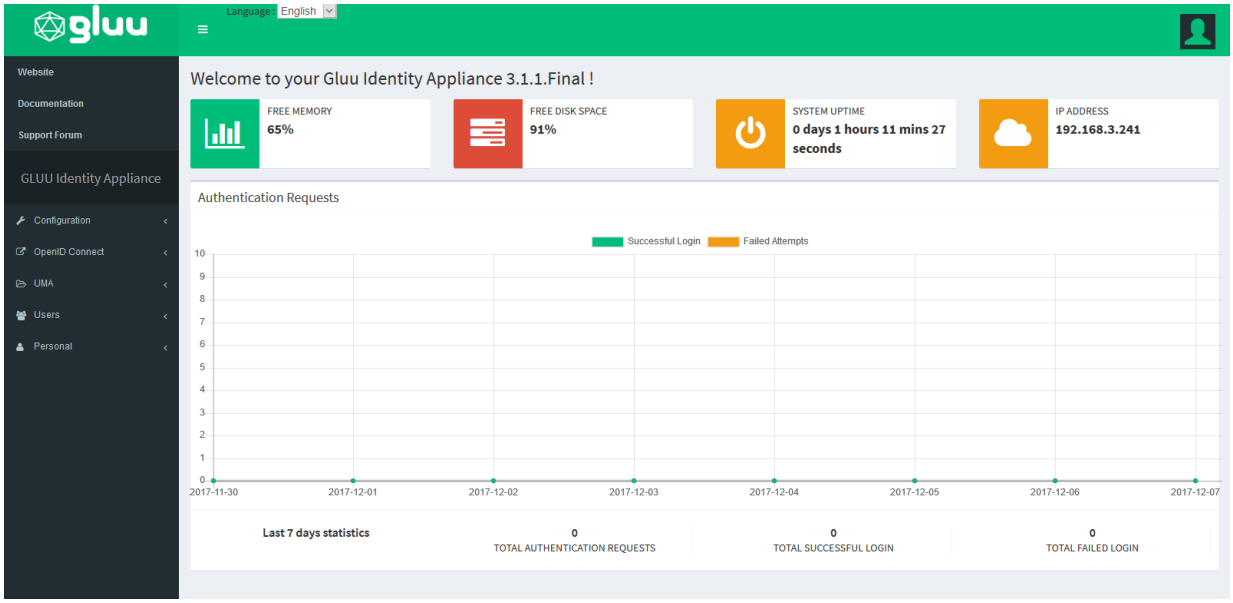

Figure 10: Gluu oxTrust Admin UI

The uninstallation of the Gluu server, can be performed by following the below commands:

```
sudo service gluu-server-3.1.1 stop

sudo apt-get remove --purge gluu-server-3.1.1

sudo rm -rf /opt/gluu-server-3.1.1.save
```

4.2.2. Data anonymization

The ARX is a cross-platform software, as it supports various operating systems. The initial virtual machine we used was composed of 4 GB of RAM and 100 GB of Hard Disk space. The Operating System, that was used, is the Ubuntu Linux 16.04.3 64bit. The hardware that we used is likely to change, depending on the load during the anonymization process. To install ARX, download and execute the installation script from the website [21]. Execute the following commands to launch ARX installer:

```
wget -O arx-3.6.0-x64.run http://arx.deidentifier.org/?ddownload=2135

chmor +x arx-3.6.0-x64.run

./ arx-3.6.0-x64.run
```

4.2.3. Trust and reputation modelling

After its clearance for being released, the source code for the trust and reputation component will be stored and made available within the central Git-based repository of the CrowdHEALTH project.

5. Conclusions

The software prototype of the Integrated Holistic Security and Privacy Framework is the main module of the CrowdHEALTH system that is responsible for the security, in a holistic way. Its' goal it so to conform with the legislation, and secure the data, and assets of CrowdHEALTH from unwanted disclosure, and malevolent usage. This is performed by combining several technologies and protocols, in order to support user authentication, authorization and access control, data anonymization, and trust management within a continuous expanding cross border system. The prototype will go under thorough testing in order to verify its functionality before the system integration.

The CrowdHEALTH consortium will stay informed about the relevant EU legislation and will modify the components of the Security and Privacy Framework accordingly to conform with the current legislation. The next revision of this deliverable is D4.21 which will include all the implementation details and changes that will be performed in the upcoming months.

References

[1] “CrowdHEALTH - D4.17: Integrated Holistic Security and Privacy Framework: Design and Open Specification v1,” November 2017. [Online].

[2] “OPenID Connect - Welcome to OpenID Connect,” [Online]. Available: http://openid.net/connect/.

[3] “OAuth 2.0,” [Online]. Available: https://oauth.net/2/.

[4] W. Yeong, T. Howes and S. Kille, “Lightweight Directory Access Protocol,” 1995. [Online]. Available: https://www.ietf.org/rfc/rfc1777.txt.

[5] “Gluu: Fast, Flexible, and Free Open Source Identity & Access Management,” 2017. [Online]. Available: https://www.gluu.org/.

[6] “The MIT License,” [Online]. Available: https://opensource.org/licenses/MIT. [Accessed 2017].

[7] “OpenID Certification,” [Online]. Available: http://openid.net/certification/.

[8] “Security Assertion Markup Language (SAML),” [Online]. Available: http://saml.xml.org.

[9] J. Brennan, “WG - User Managed Access,” [Online]. Available: https://kantarainitiative.org/confluence/display/uma/Home.

[10] “System for Cross-domain Identity Management,” [Online]. Available: http://www.simplecloud.info/.

[11] “GET STARTED WITH FIDO,” FIDO Alliance, [Online]. Available: https://fidoalliance.org/.

[12] “Single sign-on,” Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Single_sign-on.

[13] “ARX - Powerful Data Anonymization,” [Online]. Available: http://arx.deidentifier.org/.

[14] F. Prasser, R. Bild, J. Eicher, H. Spengler, F. Kohlmayer and K. A. Kuhn, “Lighthing: Utility-Driven Anonymization of High-Dimensional Data,” *Transactions on Data Privacy,* vol. 9, pp. 161-184, 2016.

[15] “ARX - Powerful Data Anonymization: Publications,” [Online]. Available: http://arx.deidentifier.org/development/publications/.

[16] CLIPS, “CLIPS: A Tool for Building Expert Systems,” [Online]. Available: http://www.clipsrules.net/. [Accessed December 2017].

[17] L. Sweeney, “k-anonymity: a model for protecting privacy,” *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems,* vol. 10, no. 5, pp. 557-570, 2002.

[18] Drools, “Drools - Business Rules Management System,” [Online]. Available: https://www.drools.org/. [Accessed December 2017].

[19] “Github - Gluu Inc.,” Gluu Inc., 2017. [Online]. Available: https://github.com/GluuFederation/.

[20] “Github - arx-deidentifier,” [Online]. Available: https://github.com/arx-deidentifier/arx.

[21] “ARX - Downloads,” [Online]. Available: http://arx.deidentifier.org/downloads/.

[22] C. Project, “D4.17: Integrated Holistic Security and Privacy Framework: Design and Open Specification v1”.