BIOBANKCLOUD
Your PaaS for Biobanking

SEVENTH FRAMEWORK PROGRAMME

Scalable, Secure Storage Biobank

Grant Agreement Number: 317871

BiobankCloud Security:
D3.3, Security Toolset (alpha version)

Final

Version:	1.5
Responsible Partner:	Ali Gholami, KTH
Date:	2015-02-26

Project and Deliverable Information Sheet

Scalable Secure Storage Biobank Project	**Project Ref. №: 317871**	
	Project Title: Scalable, Secure Storage Biobank	
	Project Web Site: http://www.biobankcloud.eu	
	Deliverable ID: D3.3	
	Deliverable Nature: Report	
	Deliverable Level: **PU**	**Contractual Date of Delivery:** 30 / September/ 2014
		Actual Date of Delivery: Resubmission
	EC Project Officer: Saila Rinne	
	Partner Responsible: Ali Gholami, KTH	
	Contributing Partners: KTH & KI & Charité	

* - The dissemination levels are indicated as follows: **PU** – Public, **RE** – Restricted to other participants, **CO** – Confidential, only for members of the project (including the Commission Services).

Document Status Sheet

Version	Date	Description	Author/Partner
0.1	2014-05-14	Initial version, TOC	Ali Gholami /KTH
0.2	2104-03-06	User administration in LDAP	Ali Gholami /KTH
0.3	2014-07-29	Two-factor authentication	Ali Gholami /KTH
0.4	2014-09-10	Authorization and conclusions	Ali Gholami /KTH
0.5	2014-09-11	General comments	Lora Dimitrova/Charité
0.6	2014-09-29	User management GUI	Jim Dowling/KTH
0.7	2014-09-30	Final version	Ali Gholami/KTH
0.8	2015-01-26	Fixed the reviewers feedback: added the conceptual architecture, general workflow execution, deviations from the D3.2 plan	Ali Gholami/KTH
0.9	2015-01-27	Comments on the modified version	Lora Dimitrova/Charité
1.0	2015-01-29	Comments on the executive summary, contents, structure, and background	Jim Dowling/KTH
1.1	2015-02-06	Comments on the modified version	Roxana Merino Martinez/KI
1.2	2014-02-12	Added background section and revised according to WP1 and WP6 comments	Ali Gholami/KTH
1.3	2014-02-12	Comments on the final draft	Lora Dimitrova/Charité
1.4	2014-02-12	Comments on the Hadoop architecture, preliminary role model, REST authentication and audit system	Jim Dowling/KTH,
1.5	2014-02-26	Final version	Ali Gholami/KTH

Contents

EXECUTIVE SUMMARY 6
1. Introduction 7
2. Background 7
2.1 Overview of BiobankCloud Platform 7
2.2 Hadoop Normal Mode 9
2.3 Hadoop Secure Mode 10
2.4 Related Security Projects in Hadoop. 10
2.4.1 Apache Rhino 10
2.4.2 Apache Knox 11
2.4.3 Apache Ranger 11
2.4.4 Apache Sentry 11
2.4.5 Kerberos used for authentication and for service-level authentications 11
3. Identification 11
4. Security Framework Architecture 13
4.1 Conceptual Architecture 13
4.2 Workflow Execution 14
5. Authentication 14
5.1 Mobile Two-Factor Authentication 15
5.2 Yubikey Authentication 16
5.3 Configuring the Yubikey Tokens 17
5.4 X509 Certificate Authentication 19
5.5 Client REST Authentication 19
6. Authorization 20
6.1 Role-Based Access Control Model 20
6.2 Per-Site Access Control 21
6.3 Per-Study Access Control 21
6.4 Credential Store 21
7. Auditing 22
7.1 Security Related Events 22
7.2 Study Management Events 23
8. User Administration 23
8.1 User Administration in LDAP 23
8.1.1 LDAP attributes, object classes and schemas 24

8.1.2 LDAP Server Administration 27
8.2 Web GUI User Administration 28
9. Discussion 29
9.1 Functional Requirements 29
9.2 Technical Requirements and Restrictions 29
9.2.1 Identification 30
9.2.2 Authentication 30
9.2.3 Authorization 30
9.2.4 Auditing 30
10. Conclusions and Future Work 30
References 31

List of Figures

Figure 1, BiobankCloud Security Architecture 8
Figure 2, Hadoop Architecture containing Yarn as Resource Manager 9
Figure 3, User Registration Form 12
Figure 4, Scanning the QR Code for New Users 12
Figure 5, Conceptual Architecture of the Security Framework 13
Figure 6, Running Workflows in the Platform by a BBC Researcher 14
Figure 7, Login Page for Users Login 15
Figure 8, Two-factor Authentication in the BiobankCloud 15
Figure 9, OTP Generated by the Google Authenticator in an iPhone 5 16
Figure 10, Generated QR Codes for User Registration 16
Figure 11, Inserting Yubikey token for configuration 17
Figure 12, Configuring Yubikey devices through Personalization tool 17
Figure 13, Configuring a Yubikey Token for a New User 18
Figure 14, BiobankCloud Login Page 18
Figure 15, Different Roles in the BiobankCloud 20
Figure 16, MySQL Relational Model to Store User/Role Credentials in the BiobankCloud 22
Figure 17, Adding Members to a Study 28
Figure 18, Changing Roles in a Study 29

List of Tables

Table 1, BiobankCloud Actors and Roles [16]. 20
Table 2, Access Control Table, Create (C), Read (R), Update (U), Delete (D), Execute (X) 21
Table 3, Study Access Control Table, Create (C), Read (R), Update (U), Delete (D), Execute (X) 21
Table 4, Security Events Audit Trails, Create (C), Read(R), Update (U), Delete (D) 23
Table 5, Study Management Events Audit Trails 23

EXECUTIVE SUMMARY

This deliverable presents the implementation of the security toolset alpha version, deliverable D3.3 for the BiobankCloud platform. The requirements for this security toolset were established from the regulatory and ethical requirements from WP1, our earlier design deliverable, and feedback from other partners. First, we present the conceptual architecture of the security framework that outlines a security architecture including authentication, authorization, auditing and user management components to provide strong security and compliance with the EU sensitive data protection directive.

The authentication component is based on the concept of two-factor authentication that is a user requires two pieces of information to login to the system using smart phones, Yubikey devices and X509 certificates. In this framework, we support a password as the first piece of info, and the second piece of info can be acquired from a smartphone app, Yubikey device or an X509 certificate. We chose these devices as they uniformly provide a pin code as the second factor of authentication and depending on the user preferences, providers of the BiobankCloud will support a flexible number of ways for users to authenticate.

The authorization component proposes role-based access control (RBAC) with extra support from the built-in Java Enterprise Edition authorization mechanisms for even more granular access control. The audit component stores for user actions and those actions are visible and searchable from user interfaces. In addition, the user management component enables the administrative users to activate/deactivate and change status of users.

In this deliverable we discuss implementation details of various services such as authenticating, authorization and user management. D3.3 provides a security toolset to be tested by the platform users. Additionally, we take into consideration feedback from users and will provide a final release that addresses new project or security requirements. We will also provide documentation and user guides to use the final release of the security toolset.

1. Introduction

This document describes the security toolset implementation for the BiobankCloud project [1] according to the design deliverable alpha version (D3.2) [10]. The ethical and legal requirements of the BiobankCloud have been discussed in [8] and [9]. The deliverable D3.2 proposed a security framework to be implemented as alpha version release.

In deliverables D3.1 [2] and D3.2, we identified the platform security requirements and designed the security framework to be implemented in D3.3 as the security toolset alpha version. The main outcomes of D3.1 and D3.2 were to find the trade-offs in the BiobankCloud project and propose a security architecture that secures the storage and processing of genomic data that contains sensitive information.

The deliverable D3.3 - security toolset alpha version, delivers the basic functionalities to ensure ***confidentiality***, ***integrity*** and ***non-repudiation*** of data access in the platform. This enables WP3 to get feedback from the BiobankCloud participants on usability of the platform and enhance the security functionalities to be included in the final toolset release (D3.4).

This document outlines the design and implementation of the security toolset including, identification, two-factor authentication, authorization and auditing. It also describes deviations from the planned design in deliverable D3.2 due to non-functional requirements and technical restrictions.

This deliverable is structured as follows. Section 2 gives a background on the BiobankCloud platform, in addition to Hadoop in default and secures modes. Section 3 explains the identification scheme that is implanted in the project. Section 4 outlines the architecture of the security framework. Section 5 discusses the authentication approaches for user authentication. Section 6 describes the authorization model. Section 7 describes the auditing system. Section 8 presents the user administration. Section 9 discusses the project requirements and deviations from the D3.2 plan. Section 10 presents the conclusions and future work.

2. Background

This section gives an overview of the BiobankCloud platform that runs the Hadoop cluster to perform the analysis of the genomic data.

2.1 Overview of BiobankCloud Platform

The BiobankCloud platform provides scalable analysis of the genomic data in terms of storage and computing power. It provides interfaces through the ***Lab Information Management System (LIMS)*** for users to upload genomic data and use the BiobankCloud services as shown in Figure 1.

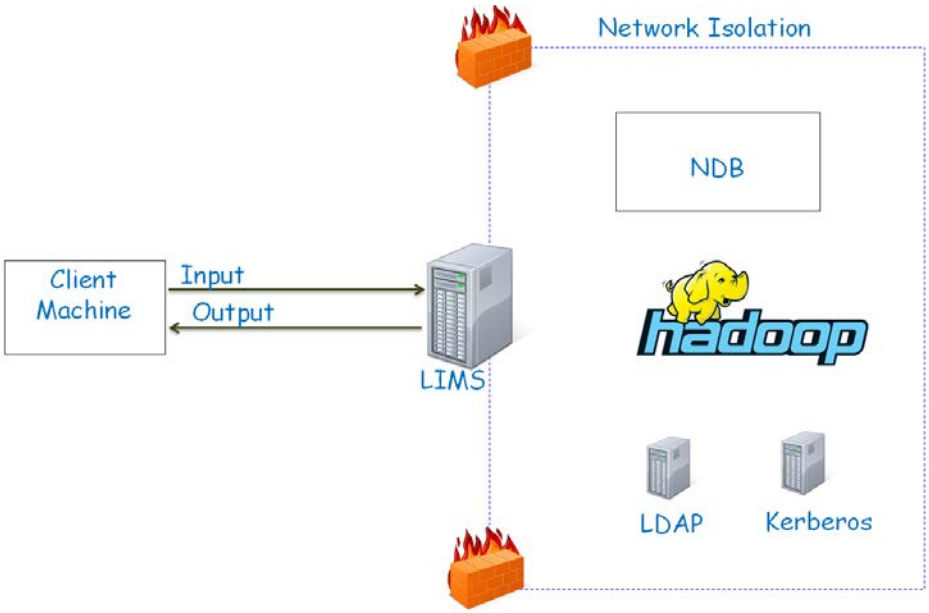

Figure 1, BiobankCloud Security Architecture

The LIMS receives user tasks and submit them to a Hadoop cluster where Hadoop[1] distributed file system (HDFS) have multiple metadata servers, consisting of NameNodes and network database (NDB)[2] nodes (that store the actual meta-data). NDB nodes are the nodes that make up MySQL Cluster. The user credentials will be stored in a relational database, LDAP [18] and Kerberos [19] servers.

All BiobankCloud services are protected with firewall and therefore only secure connections are allowed to access the platform. User initiates a job submission through the client machine and LIMS process the request for authorized access to the platform.

The Hadoop cluster runs the submitted job and will notify the user through the LIMS interfaces. Hadoop is composed of two main components: HDFS and Map Reduce. HDFS is a distributed file system that keeps the metadata in NameNode server as a hierarchical file and directory name space. The NameNode server also stores the DataNodes that keep track of the individual blocks of each file [15], as shown in Figure 2. The NameNode uses remote procedure call (RPC) protocol and read/write to DataNodes using a streaming socket protocol called the data-transfer protocol.

MapReduce is another key component of Hadoop framework for processing large amounts of data in parallel using the MapReduce [17]. A user submits a MapReduce job to the JobTracker to be put in the queue. The job will be executed by the compute nodes, as shown in Figure 2. Each compute node contains a TaskTracker that performs map and reduce for each job.

[1] Hadoop, http://hadoop.apache.org/
[2] MySQL NDB, http://dev.mysql.com/doc/refman/5.1/en/mysql-cluster.html

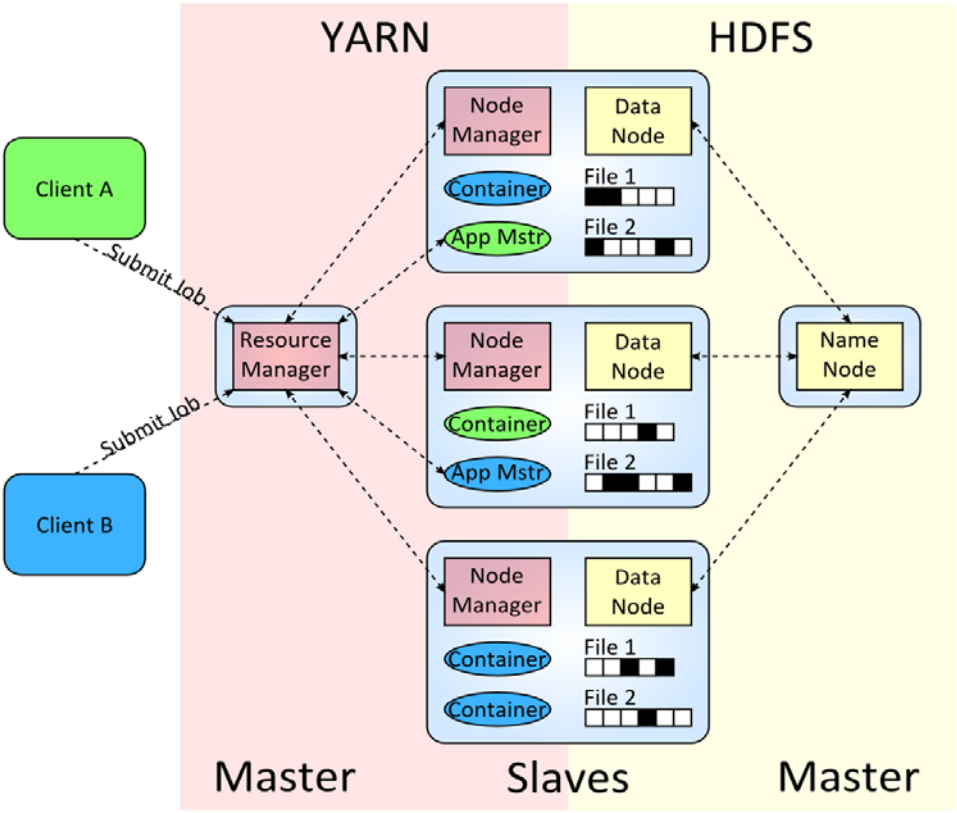

Figure 2, Hadoop Architecture containing Yarn as Resource Manager

Hadoop has been used by many organizations without demand for strong security [15]. Only few companies have deployed secure Hadoop environments such as Yahoo!. Therefore Hadoop built-in security requires tailoring for different security requirements. Hadoop operates in two modes: normal (non-secure) and secure modes.

2.2 Hadoop Normal Mode

Hadoop default configurations are in non-secure mode. The default mode has no authentication enforcement [20]. It relies on client-side libraries to send the credentials from the user machine operating system in context of the protocol [15]. Clusters are usually deployed onto private clouds with restricted access to authorized users.

In this model, all users and programmers have similar access rights to all data in HDFS. Any user that submits a job could access any data in the cluster and reads any data belonging to other users. Moreover, MapReduce does not authenticate or authorize submitted tasks. An adversary is able to tamper with the priorities of other Hadoop jobs in order to make his job complete faster or terminate other jobs [29].

Data confidentiality and key management are also missing in the Hadoop default mode. There is no encryption mechanism deployed to keep data confidential in HDFS and MapReduce clusters. For scenarios where confidentiality is a requirement other distribution of Hadoop can solve this issue through encryption.

2.3 Hadoop Secure Mode

Security features of Hadoop consist of authentication, service level authorization and authentication for Web consoles [20]. By configuring Hadoop in secure mode, each user and service require authentication by Kerberos in order to use Hadoop services. Since Hadoop requires a userid string to identify users, a POSIX-compliant username can be used for authentication purposes. This usernames can also be used during authorization to check the access control lists (ACL). Additionally, Hadoop supports the notion of POSIX groups to allow a group of users to access HDFS resources. Authorization checks through ACLs and file permissions are still performed against the client supplied user id.

RPC library is used to provide clients secure access to Hadoop services through sending username over simple authentication and security layer (SASL). SASL is built on Kerberos or DIGEST-MD5. In Kerberos mode, users acquire a ticket for authentication using SASL for mutual authentication. DIGEST-MD5 mechanism uses shared symmetric keys for user authentication with servers to avoid overheads of using a key distribution center (KDC) as a third party for authentication. RPC also provides data transmission confidentiality between Hadoop services clients through encryption in contrast to the Web-console that utilized SSL (HTTPS).

HDFS services either use RPC connection from the client to the NameNode or block transfer from the client to DataNodes. The RPC connection can be secured via Kerberos authentication or delegation tokens. Delegations tokens are used for simplicity and avoiding user passwords over the network. During block transfer from the client to DataNodes, the block transfer is authenticated using block access tokens.

MapReduce utilizes Kerberos authentication over RPC from the client to JobTracker. MapReduce enforces authorized access through usernames since job directory is located under user's home directory. In scenarios, where a job requires accessing multiple home directories in HDFS, user credentials are stored in Map with string keys and binary values. JobTracker is the component that stores user credentials in HDFS system directory.

2.4 Related Security Projects in Hadoop.

Recently, many organizations who realized the importance of security in their enterprises decided to add support for different security requirements. In this section we outline several efforts or approaches that aim to enhance Hadoop security.

2.4.1 Apache Rhino

Apache Rhino is an initiative started by Intel at the beginning of 2013, to remarkably enhance the current Hadoop ecosystem security. It aims at providing a framework for Hadoop key management, authorization, audit and logging [21]. Apache Rhino main contributions are as follows:

- Framework support for encryption and key management
- A common authorization framework for the Hadoop ecosystem
- Token based authentication and single sign on
- Extend HBase support for ACLs to the cell level
- Improve audit logging

2.4.2 Apache Knox

The Apache Knox (Gateway) provides perimeter security for confidential access to Hadoop clusters through organizational policies within enterprises [27]. Apache Knox enhances the Hadoop security through simplifying users' access to the cluster data and job execution.

Client interactions are performed through representational state transfer (REST) Web services over HTTP. Apache Knox also aims to provide easy integration with existing identity providers and abstracting Kerberos authentication. This is done through encapsulating Kerberos to eliminate the need for client software or client configuration of Kerberos by clients. In addition, it provides integration with security assertion mark-up language (SAML) [23], open authorization (OAuth) [24] and OpenID [25]. The Gateway dispatches REST/HTTP calls to different components of Hadoop environments.

2.4.3 Apache Ranger

Apache Ranger proposes a framework for data security across the Hadoop platforms to enable enterprises run multiple workloads in a multi-tenant environment [26]. Apache Ranger aims to provide centralized security administration to manage all security related tasks in a central user interface (UI) or using REST APIs. Fine grained authorization for specific operations through a central UI is another goal of Apache Ranger. Support for RBAC and attribute-based access control (ABAC), in addition to centralized auditing services are among the functionalities of this software.

2.4.4 Apache Sentry

Apache Sentry is another effort for securing Hadoop ecosystem through enforcing fine-grained role based authorization for data and metadata located in a Hadoop cluster [28]. Sentry implements a policy provider to define the access control. This is done through defining a single global policy file can be used to control access.

2.4.5 Kerberos used for authentication and for service-level authentications

Kerberos can be used for user authentication in Hadoop secure deployments over SSL. For organizations that require other security solutions not involving Kerberos, this demands setting up a separate authentication system. Hadoop implements SASL/GSSAPI for mutual authentication of users with Kerberos, running processes, and Hadoop services on RPC connections [29].

A secure deployment requires Kerberos settings where each service reads authentication information saved in keytab file with appropriate permission. A keytab is a file that contains pairs of Kerberos principals and encrypted keys. Keytab enable services in Hadoop to use services without being prompted to enter password for authentication.

3. Identification

We implemented a portable operating system interface (POSIX)-compliant username scheme containing 8-alphanumeric characters for compatibility with the Hadoop open platform as a service (Hops) [13]. Every time a new user registers using the self-service component, a new username will be generated and stored in the backend credential store.

The username scheme is composed of 3 letters that indicate the prefix of the institution name, and the other 5 digits are the user identifiers. For example meb10003 demonstrates a user from the MEB

institute with id number 10003. This scheme provides flexibility of user creation and integration with other institutions in cases where federation is required, in addition to high compatibility with the Hops.

The platform also supports the user ORCID identifiers [3] as a replacement for the POSIX-based usernames creation. The identity store keeps mapping of ORCID identifiers and POSIX usernames for any required queries. Figure 3 shows the user registration page for a new user account request. A user opens the registration page and enters the personal and organization information in addition to a plain password.

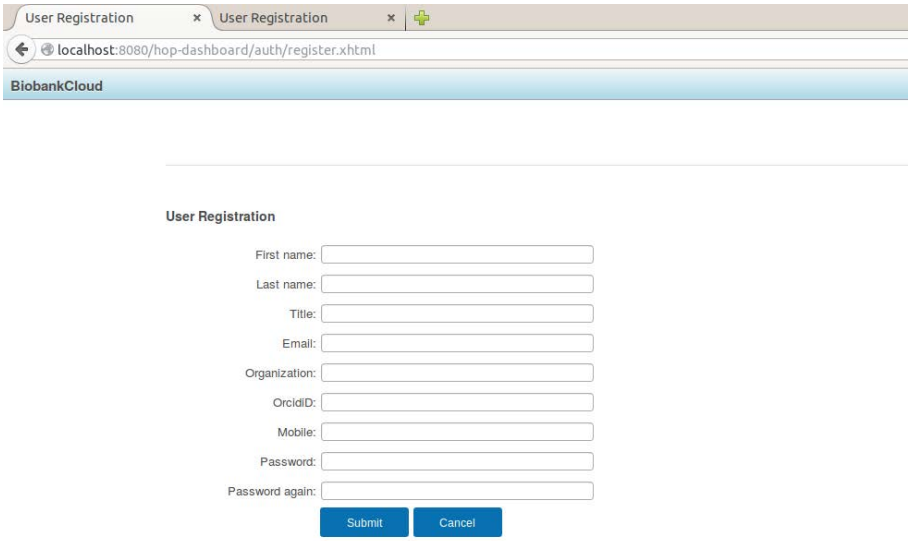

Figure 3, User Registration Form

After successful registration of the information a unique username will be created to be sent through the browser as a quick response (QR)[3]. The user will scan this code by the mobile device, as shown in Figure 4.

Figure 4, Scanning the QR Code for New Users

[3] Quick Response Code, http://en.wikipedia.org/wiki/QR_code

4. Security Framework Architecture

This section outlines the security architecture of the platform to provide confidentiality, integrity and non-repudiation of data access through different components.

4.1 Conceptual Architecture

The security toolset is composed of three main layers: client machine, application server, and credential server, as shown in Figure 5.

- **Client Machine:** User establishes secure connections through the hypertext transfer protocol secure (HTTPS) [4] to the application server through the client machine. We aim to support three groups of users: mobile users, certificate users and Yubikey users.

- **Application Server:** Application server runs the security components to handle authentication, authorization, auditing and user management requests.

- **Credential Server:** This layer contains the user credentials such as usernames, passwords, and roles, audit trails that are stored in OpenLDAP[5], MySQL[6] cluster and public-key infrastructure (PKI) key store.

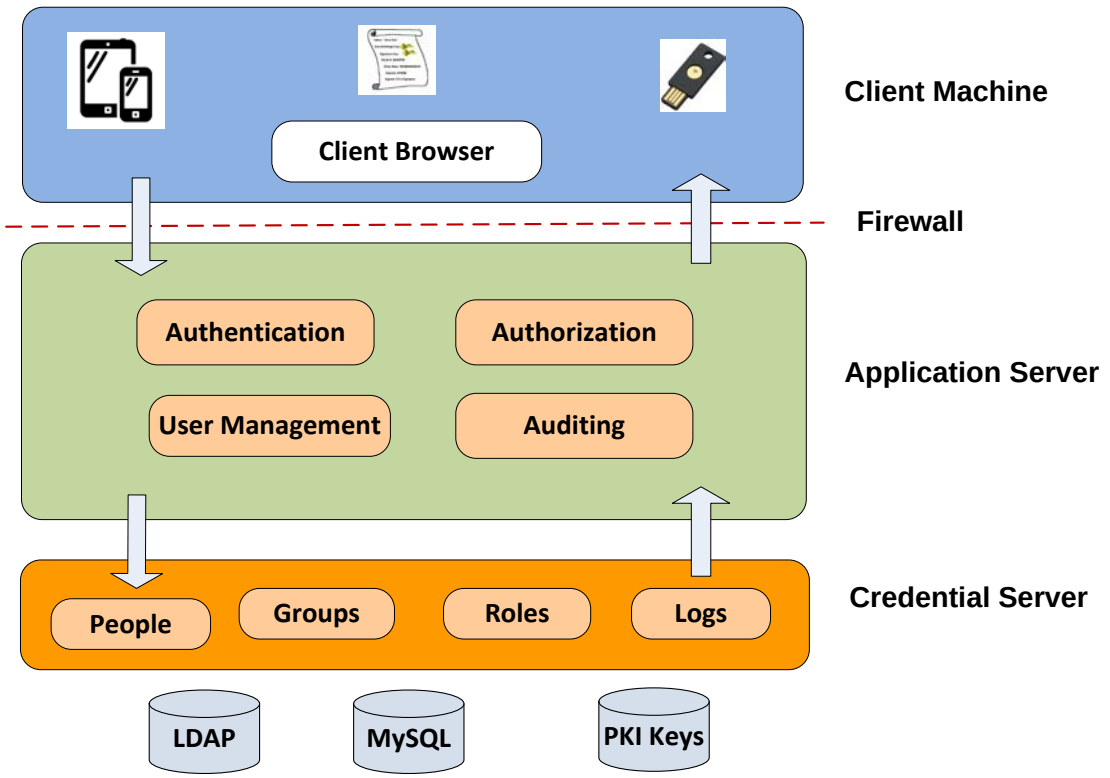

Figure 5, Conceptual Architecture of the Security Framework

[4] Hypertext Transfer Protocol Secure (HTTPS), http://en.wikipedia.org/wiki/HTTP_Secure
[5] OpenLDAP, http://www.openldap.org/
[6] MySQL, http://www.mysql.com/

4.2 Workflow Execution

A typical scenario to authenticate and authorize users in the platform is depicted in Figure 6. The BiobankCloud researcher (BBC Researcher) accesses the platform in following steps:

1. User opens the login page and enters the authentication credentials.
2. Authentication component verifies identity of the user through checking the People credential database.
3. If the user is authenticated, access to platform services for the BBC Researcher will be provided.
4. BBC Researcher runs a workflow in the platform.
5. Authorization component ensures that the BBC Researcher has enough permission to run the workflow.
6. Depending on the permission check:
    a. If BBC Researcher is not authorized to run the workflow he/she will be denied,
    b. If BBC Researcher is granted to access the asset he/she will be permitted.
7. Execution engine in the BiobankCloud runs the workflow and access the genomic data store which is located in Hops and presents the results to the BBC Researcher.

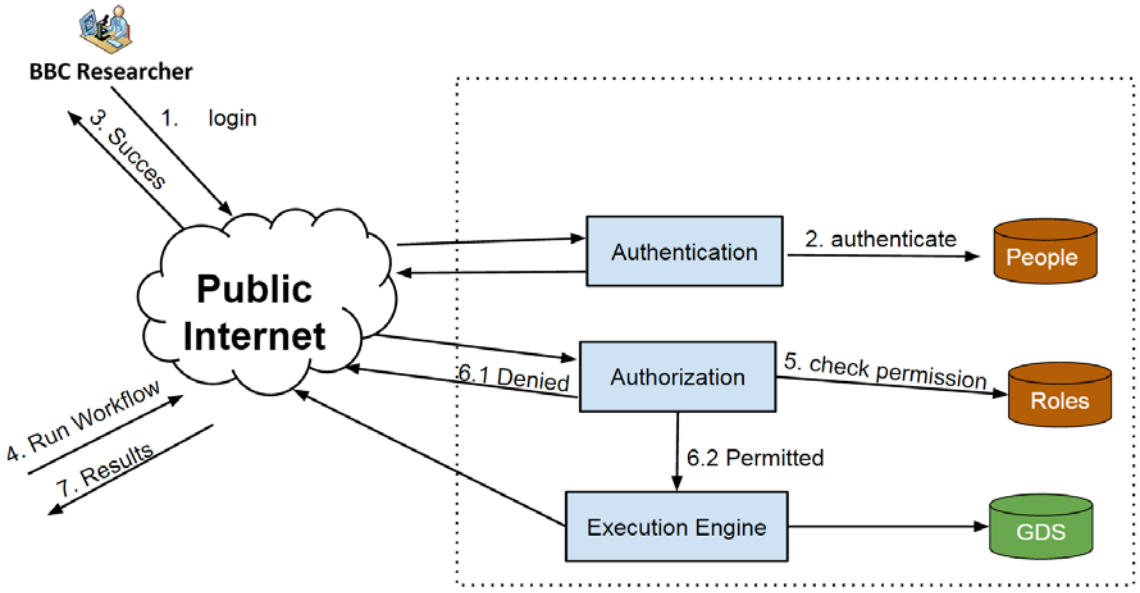

Figure 6, Running Workflows in the Platform by a BBC Researcher

5. Authentication

Requirements of using strong security mechanisms in the BiobankCloud have been discussed in the deliverables D3.1 and D3.2. We aim to support three groups of users: users with smart mobile devices, users with Yubikey tokens and admin users with certificates.

The motivation for this categorization stems from the fact that not all researchers use mobile devices for authentication neither they all use Yubikey tokens. Also authentication using certificates ensures more security for admin roles. However, this is not a hard requirement and we will consider integrating certificate authentication in the final release.

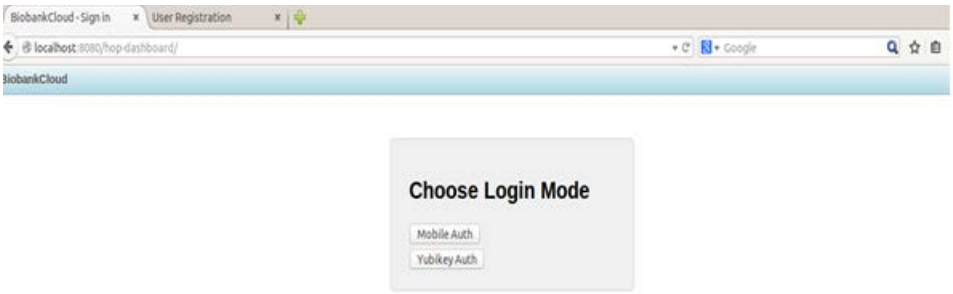

Figure 7, Login Page for Users Login

In this project we developed two-factor authentication using time-based one-time password (TOTP) [4] for mobile devices and keyed-hash message authentication code (HMAC)-based one-time password (HOTP) [6] for Yubikey tokens. Users will decide the login method in the login page as shown in Figure 7.

The BiobankCloud users supply the one-time password (OTP) in addition to the static passwords which are established during account creating. Figure 8 shows the design of the two-factor authentication system in the application server layer.

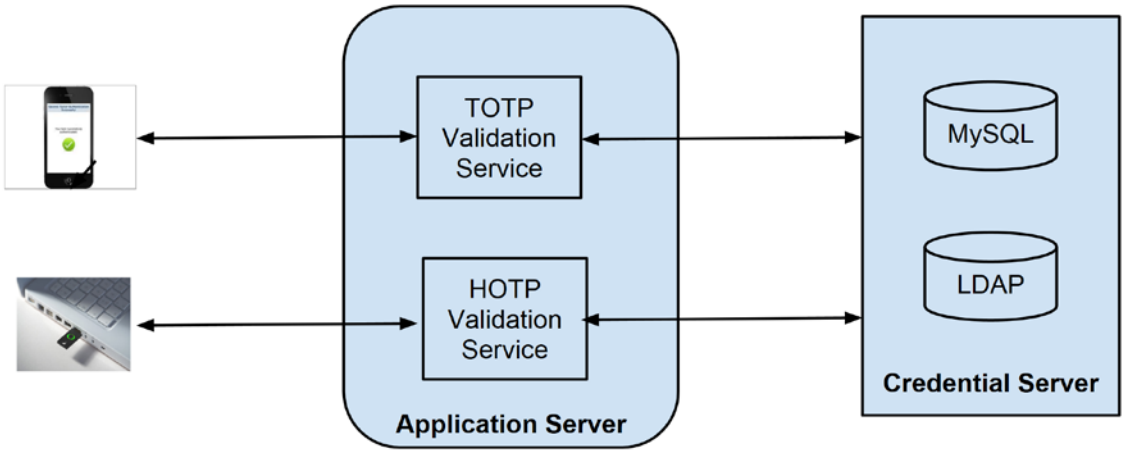

Figure 8, Two-factor Authentication in the BiobankCloud

Mobile two-factor authentication requests are sent to the TOTP validation service while Yubikey authentication requests are sent to the HOTP validations service. Each validation service retrieves the credentials from the credential server. We implemented two credential stores using LDAP and MySQL.
The LDAP service provides the user management through command-line, while the MySQL database is used for graphical user interface (GUI) support. The credential server requires implementation of provisioning and synchronization components to keep both LDAP and MySQL synchronized together.

5.1 Mobile Two-Factor Authentication

Google Inc. have implemented TOTP (RFC 6238) [4] security tokens in terms of Google Authenticator [5]. As shown in Figure 9, the Google Authenticator generates one-time passwords (six digits) that users must provide in addition to their username and password to log into Google services

or other sites. The TOTPs are generated in 30 second periods. We support two common mobile devices platforms: iOS and Android. However, there might be new platform supports for Goolge Authentication for other mobile devices.

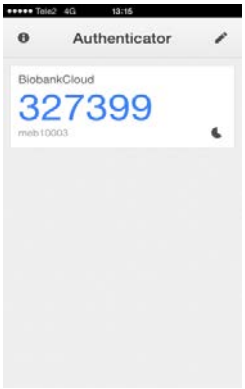

Figure 9, OTP Generated by the Google Authenticator in an iPhone 5

For mobile authentication, we implemented a customized QR code library that contains usernames and their association information as displayed in Figure 10. Users only require to locate their mobile devices towards the generated QR code by the BiobankCloud and all the account information will be loaded automatically to their devices. The QR codes are presented in portable network graphics (PNG) with the size of 200 * 200 pixels.

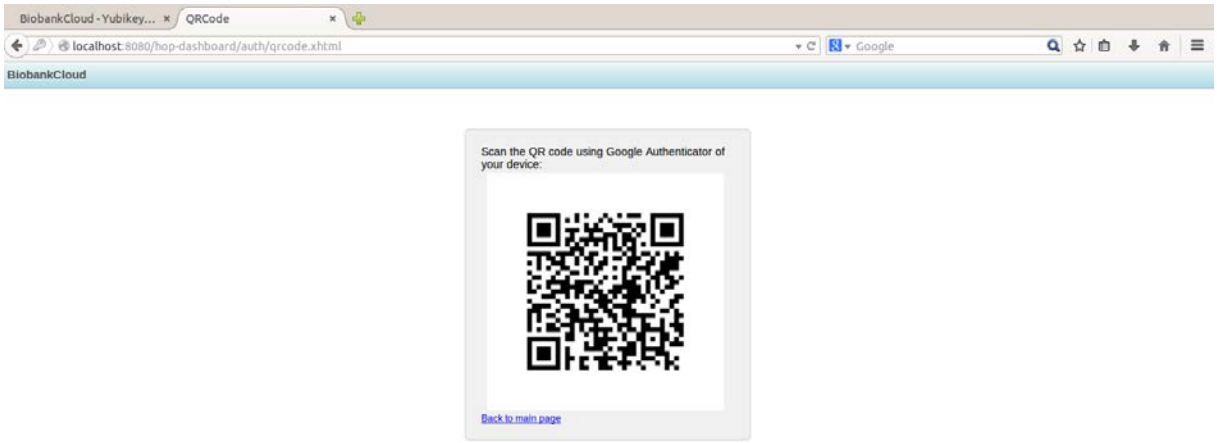

Figure 10, Generated QR Codes for User Registration

5.2 Yubikey Authentication

We implemented the RFC4226 [6] specification to support Yubikey authentication. We also implemented a validation service in the platform to verify the issued OTPs presented by users. A Yubikey token generates the OTPs through a push-button. Generated OTPs are sent as emulated keystrokes via the keyboard input path, thereby allowing the OTPs to be received by any text input field in the authentication page of the platform.

To generate an OTP, at first the user inserts the Yubikey into the USB port as shown in Figure 11. The user presses the Yubikey's OTP generation button. The Yubikey generates an encrypted string of characters that are outputted as keystrokes via the keyboard port. This output will be redirected to the OTP field of the authentication page.

Figure 11, Inserting Yubikey token for configuration

The platform users are required to supply an additional static password and then press the login button to be authenticated. The Yubikey validation service is triggered and it verifies the claimed user credentials presented as a string through the Web. Our validation service converts the received string to a byte string to be decrypted using the same (symmetric) 128-bit AES[7] key.

The AES secret will be fetched from the credential stores. Then the string's checksum will be checked and if not valid, the OTP will be rejected. Additional fields will be checked as next step and if not valid, the OTP will be rejected. As next step, the non-volatile counter will be compared with the existing value in the credentials store. If lower than or equal to the stored value, the received OTP will be rejected. If greater than the stored value, the received value is stored and the OTP will be validated.

5.3 Configuring the Yubikey Tokens

To configure the Yubikey, personalization GUI software should be installed. The command "apt-get install yubikey-personalization-gui" installs the required dependencies and it can be invoked as shown in Figure 12.

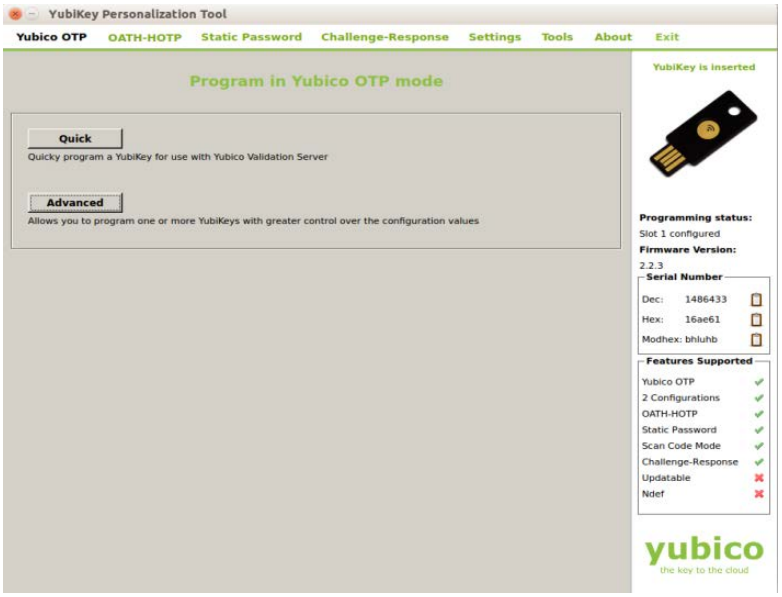

Figure 12, Configuring Yubikey devices through Personalization tool

[7] Advanced Encryption Standard, http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

The credentials can be written in one of the two configuration slots as shown in Figure 13. We use open authentication (OATH)[8] standard token identifier with 6 bytes length and also HOTP of 6 bytes length.

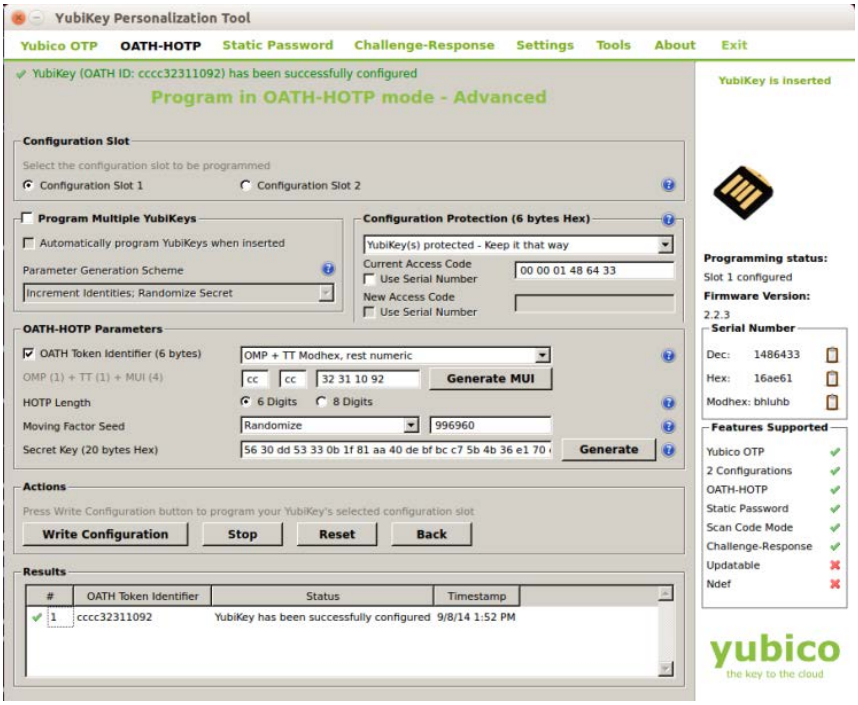

Figure 13, Configuring a Yubikey Token for a New User

After selecting the moving factor seed as "Randomize", the "Write Configuration" action should be selected to write the configurations in both a CSV file and the actual Yubikey device. The identity store then can be updated with the following credentials:

```
OATH-HOTP, 9/8/14 2:02 PM, 1, ccccedebbckd, , 5630dd53330b1f81aa40debfbcc75b4b36e170ed,
000001486433,000001486433,0,1,0,6,996960,0,0,0,0,0
```

Figure 14 present s the login page for Yubikey two factor authentication, where a user requires to enter username, password and OTP through the Yubikey device.

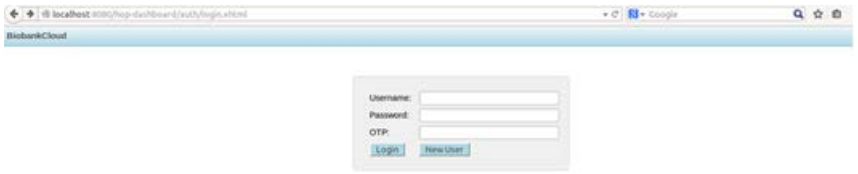

Figure 14, BiobankCloud Login Page

[8] OATH, http://www.openauthentication.org/

5.4 X509 Certificate Authentication

The BiobankCloud administrative staff, such as the data controller, is required to login to the platform services using public key certificates. For this purpose, the BiobankCloud services will be identified to the clients through a server certificate and also clients require a valid certificate to communicate with each other.

The certificates are stored in the Java key store and obtaining and installing a valid certificate is done through the "keytool" command in GlassFish v3 [7]. The client distinguished name (DN) is required to be embedded in the web.xml file of the Web application.

```
<login-config>
        <auth-method>CLIENT-CERT</auth-method>
</login-config>

<security-role-mapping>
       <role-name>CONTORLLER</role-name>
       <principal-name>: CN= Ali Gholami nospam@kth.se, O=Kungliga Teknikska
       Hogskolan,C=SE,DC=TCS,DC=Terena,DC=org </principal-name>
</security-role-mapping>
```

The users are then required to import the certificate to their browser when using the platform that is also protected through a password.

5.5 Client REST Authentication

The BiobankCloud security architecture provides flexibility to be extended with REST API, to support client authentication. REST authentication is performed against a HTTP server that every operation has a unique URL.

The URL contains the following information:

- Operation: Authentication
- Operation URL: https://host:port/security/authenticate
- Parameters: username, password, OTP, mode, uri
- Output: authentication token

After successful authentication, users' machine caches the authentication tokens for authorization when accessing the platform services. This includes an authorization operation:

- Operation: Authorization
- Operation URL: https://host:port/security/authorize
- Parameters: uri, action, authentication token
- Output: Permitted or denied based on the user's requested action over the resource

6. Authorization

This section describes the authorization system in the BiobankCloud project including the role model in Table 1 [16] and study management.

Actor	Roles
Researcher	Controller Trusted researcher Guest
Platform	Administrator Processor Auditor Access committee
Ethics Board	Auditor Guest
Organization	Guest

Table 1, BiobankCloud Actors and Roles [16].

6.1 Role-Based Access Control Model

Figure 15 shows a high level view of the authorization model containing the BiobankCloud roles. For avoiding the complexity of the legal terminology buy the platform users we map the roles defined in Table 1 as below:

- BBC Guest: users who create a new account request to be approved.
- BBC User: users with approved account status that are able to upload genetic data to the platform
- BBC Admin: users who have created a study and uploaded data to the platform. This group is able to add or remove other researchers to the created studies
- BBC Researcher (Trusted Researcher): general users of the platform that are trusted to upload new data or use the permitted services.
- SYS Admin: users with the possibility to provide, entitle or revoke access to other groups.
- Auditor: users with only "read" access to audit trails and user activities.
- Ethics Board: group of users with auditor and BBC researcher roles.

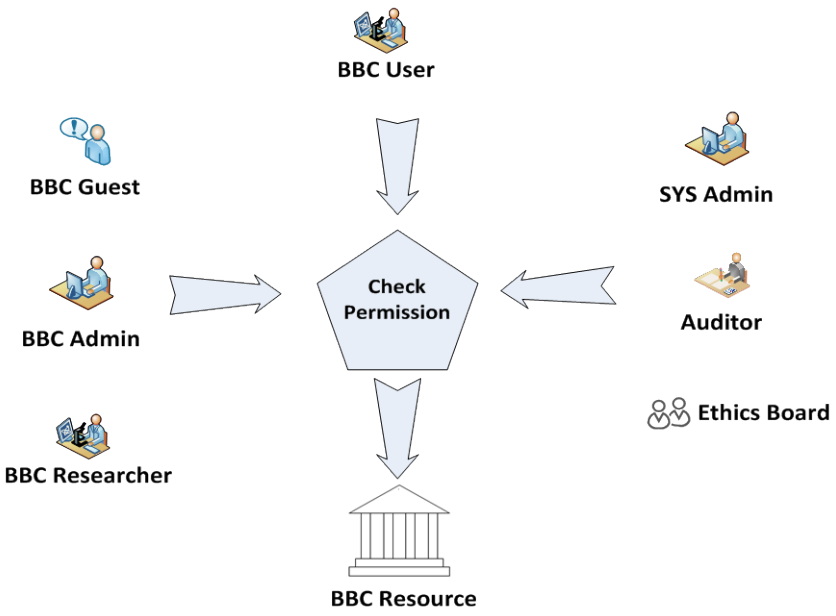

Figure 15, Different Roles in the BiobankCloud

We introduce BBC Admin as ***Controller*** and SYS Admin as ***Processor*** for simplicity and avoiding the legal terminology. This model merges similar roles and actors for usability and simplicity in the implementation. For instance we used the concepts of controller and processor roles to be implemented in BBC Admin and BBC Researcher roles. This model will be tested and finalized in the deliverable D3.4.

6.2 Per-Site Access Control

Table 2 shows different roles and related access rights including create, read, update, delete and execute per-site in the platform. Each role, such as SYS Admin, Auditor, Guest Researcher, Trusted Researcher, BBC User, Ethics Board. Each role has different permissions to access audit management, data anonymization service, platform public pages and user administration services.

Role / Service	SYS Admin	Auditor	Guest Researcher	Trusted Researcher	BBC User	Ethics Board
Audit Management	R	R			R	R
Data Anonymization Service				X		X
Platform Public Pages	C,R,U,D	R	R	R	R	R
User Administration	C,R,U,D	R				R

Table 2, Access Control Table, Create (C), Read (R), Update (U), Delete (D), Execute (X)

6.3 Per-Study Access Control

Table 3 shows different roles and related access rights including create, read, update, delete and execute for study management. Study management functionalities include creating/removing studies, adding and assigning roles to a study as team member, browsing studies, running Cueiform jobs and accessing the Cueiform job results.

Role / Service	SYS Admin	Auditor	BBC Admin	Trusted Researcher	BBC User
Create Study			C,R,U,D		
Remove Study			X		C
Team Members		R	R,U,D	C,R	
Browse Studies	R,U,D	R	C,R,U,D	R	C
Cuneiform Jobs			C,R,U,D	C,R,U,D	
Cuneiform Job Results		R	R,U,D	R,U,D	R

Table 3, Study Access Control Table, Create (C), Read (R), Update (U), Delete (D), Execute (X)

6.4 Credential Store

To protect the confidentiality of the genomic data and platform services we implanted a relational table “Resource”, which includes the information about a specific resource such as a directory containing the sample studies.

All users are assigned a role when their account is activated. When users upload sample data sets, they can decide which permissions to assign to which users (read, write and execute). This is done in the “Permission” table.

User information such as username, email, password, registration date are stored in the People table. Group information (group name, group id, description) are kept in the Group table. User permissions to resources are kept in the Role table. Yubikey table stores the Yubikey credentials to authenticate users. Figure 16 shows the important relations that are used for authentication/authorization in the BiobankCloud. This model will be finalized in the deliverable D3.4 to include audit trail relations and enhance any possible missing features.

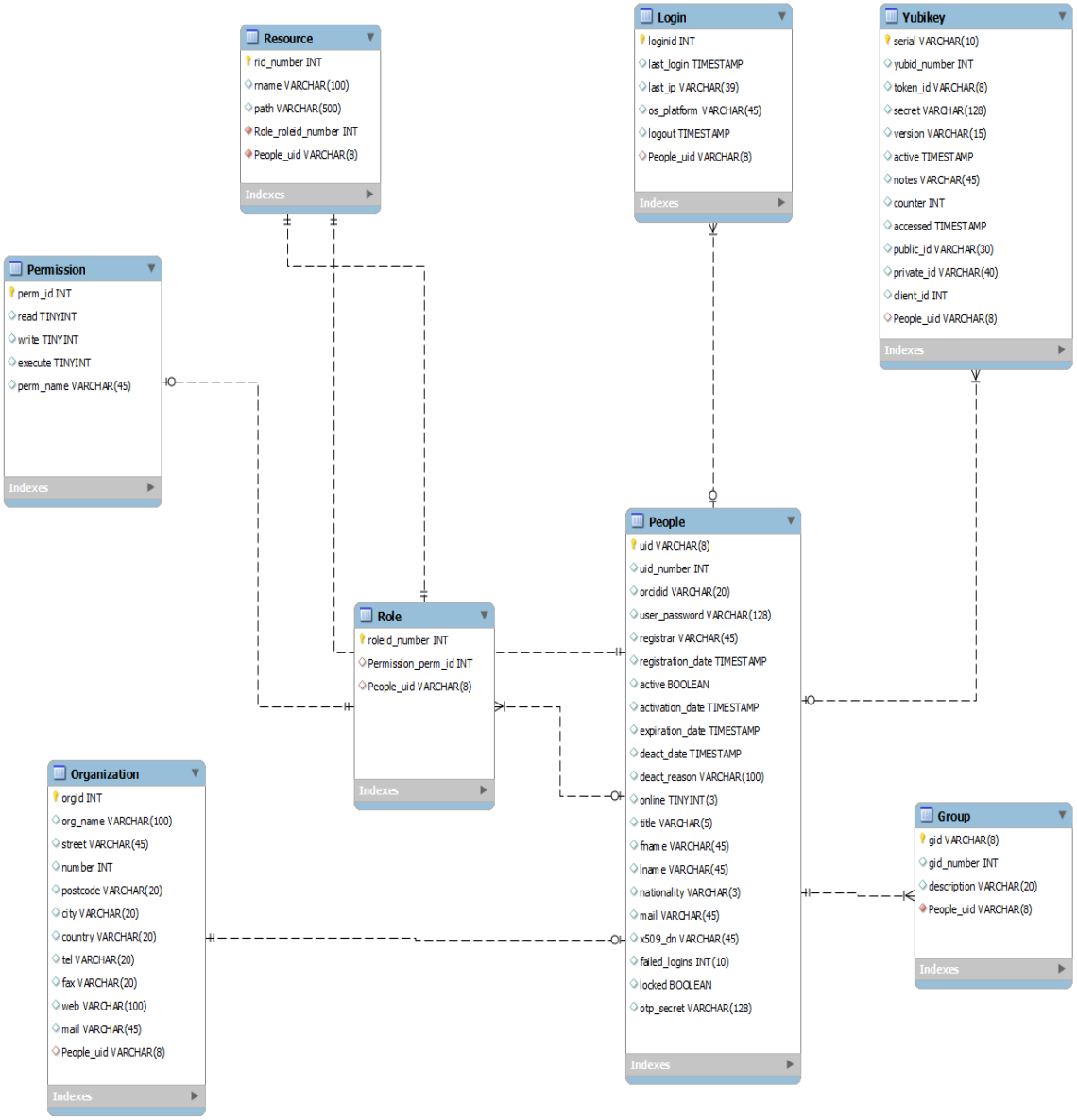

Figure 16, MySQL Relational Model to Store User/Role Credentials in the BiobankCloud

7. Auditing

Audit component generates and stores the log events related to security (user management, authentication, authorization, privacy management) and study management, as descried in the following.

7.1 Security Related Events

Security related events such as user management, authentication, authorization, privacy management, consent control, profile management and audit management logged and stored in the audit trail

database. Table 4 shows levels of logging that are required to be collected and stored for each resource.

Audit trails / Resource	User	Action	Role	Timestamp	IP address	Operating System	Browser	Outcome
User Management	✓	C,R,U,D	✓	✓				✓
Authentication	✓		✓	✓	✓	✓	✓	✓
Consent Control	✓	C,R,U,D	✓	✓	✓	✓	✓	✓
Profile Management	✓	C,R,U,D	✓	✓	✓	✓	✓	✓
Audit Management	✓	C,R,U,D	✓	✓				✓

Table 4, Security Events Audit Trails, Create (C), Read(R), Update (U), Delete (D)

7.2 Study Management Events

Study management events are audited in the platform as defined in Table 5. Each service such as study data, team member management, browsing studies, running Cuneiform jobs, study related data, and samples are logged and stored in the audit trail database.

Audit trails / Resource	User	Action	Role	Timestamp
Study Data	✓	C,R,U,D	✓	✓
Team Members	✓	C,R,U,D	✓	✓
Browse Studies	✓	R,D	✓	✓
Cuneiform Jobs	✓	C,R,U,D	✓	✓
Cuneiform Jobs Results	✓	R,D	✓	✓
Privacy Management	✓	C,R,U,D	✓	✓
Study info	✓	C,R,U	✓	✓
Samples	✓	C,R,U,D	✓	✓

Table 5, Study Management Events Audit Trails

8. User Administration

The BiobankCloud user administration service will be built on top of existing local administrations in different institutions and biobanks, and, hence should make no assumption about these systems. This will ensure each BiobankCloud user management system has minimum adaptation and interruption to the local systems, in addition to performing user management locally by each institution.

8.1 User Administration in LDAP

To enable access to the BiobankCloud services attributes of the user have to be known by that BiobankCloud. For instance, POSIX attributes such as username, user identifier (UID), group identifier (GID) and the subject name of a X.509 certificate for some users with administrative privileges.

We implemented our directory service based on the lightweight directory access protocol (LDAP) for user administration in the BiobankCloud. LDAP provides a suitable protocol for excessive lookups compared to relational databases that are transactional.

Moreover, LDAP can be used as a protocol for communication and can be integrated with several high availability databases such as the MySQL cluster. The current setup consists of distributed LDAP servers for each BiobankCloud and each platform maintains the user information in a local LDAP server.

The LDAP servers host directory entries with suffixes on each platform such as "dc=biobankcloud,dc=eu", "dc" stands for domain component that can be different for each BiobankCloud.
The account creation in the BiobankCloud consists of two phases. First a BiobankCloud manager or administrator creates the user account in the LDAP. Second, the provision system updates the execution engine of the BiobankCloud platform (Hadoop cluster) through a provision service.

Security of the LDAP server is a major concern, and to ensure confidentiality, integrity and non-repudiation we use encryption tunnels through transport layer security (TLS) provided by X.509 certificates.

Therefore, clients and administrative users will interact authentically using the SASL EXTERNAL mechanism. SASL provides the functionality of mapping certificate subjects to LDAP entries for authorization.

To forbid unauthorized access to the LDAP service, a list of authorized servers with their certificate subject (distinguished name) is defined in the LDAP server. Also, we assume the LDAP service runs in a dedicated host with enough security measures that is not accessible to irrelevant programs and processes.

8.1.1 LDAP attributes, object classes and schemas

The attributes that are defined in this section, are defined based on the requirements of WP1 and WP5. However the standard attributes of LDAP were not sufficient to address the BiobankCloud requirements, and hence we defined a schema containing attributes and objectClass definitions, as described in this section.

- POSIX passwd entry: uid (login name), uidNumber, gidNumber, gecos, homeDirectory, login shell.
- POSIX group: name, gidNumber, members (memberUid).
- Title: Mr., Mrs., Ms., Miss, Dr., and Prof.
- Home organisation of the user: bbcHomeInstitute.
- Email address: mail.
- Telephone number: telephoneNumber.
- Nationality of user: bbcNationality.
- Address and telephone number of the BiobankCloud.
- The Subject Name of the admin's certificate: The subject name (DN) of the admin users.
- Account status and deactivation reason: bbcDeactivated, bbcDeactReason.
- Name of the administrator or manager that registered the user: bbcRegistrar.
- Expiration date of the account: shadowExpire with yyymmdd or -1 values.
- Orcid Identifier: orcidID.
- Yubikey device identifier: yubikeyId.
- Yubikey static password: passwordFactor.
- Group description: Trusted Researcher, Guest Researcher, Data Provider, Auditor, Ethics Board
- Project Manager: bbcProjectManager.
- Resource: bbcProjectResource.

The actual implementation of the LDAP service, including top level entry of organization, project, group, resource and people.

```
#top level structure entry
dn: dc=biobankcloud,dc=eu
objectclass: top
objectclass: dcObject
objectclass: organization
o: biobankcloud
dc: eu

# top level organization entry
dn: ou=Organization,dc=biobankcloud,dc=eu
objectClass: top
objectClass: organizationalUnit

# top level user entry
dn: ou=People,dc=biobankcloud,dc=eu
ou: People
objectClass: top
objectClass: organizationalUnit

# top level group entry
dn: ou=Group,dc=biobankcloud,dc=eu
ou: Group
objectClass: top
objectClass: organizationalUnit

# top level group entry
dn: ou=Project,dc=biobankcloud,dc=eu
ou: Project
objectClass: top
objectClass: organizationalUnit

# top level resource entry
dn: ou=Resource,dc=biobankcloud,dc=eu
ou: Resource
objectClass: top
objectClass: organizationalUnit

# project entry
dn: cn=STHLM2,ou=Project,dc=biobankcloud,dc=eu
cn: STHLM2
bbcHomeInstitute: KI
bbcProjectEndTimestamp: 20140101235959Z
bbcProjectStartTimestamp: 200501010000000Z
description: STHLM2
gidNumber: 90008
memberUid: meb00001
objectClass: top
objectClass: posixGroup
```

objectClass: bbcProject
bbcProjectManager: meb00001

user entry
dn: uid=meb00001,ou=People,dc=biobankcloud,dc=eu
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
objectClass: bbcUser
cn: Ali Gholami
gecos: Ali Gholami
givenName: Ali
sn: Gholami
mail: nospam@pdc.kth.se
telephoneNumber: +46 71 234500
title: Mr.
uid: meb00001
uidNumber: 1300001
gidNumber: 1300001
loginShell: /bin/bash
shadowExpire: -1
bbcNationality: SE
homeDirectory: HDFS
userpassword: 438ruifneu38ebn23r832..
orcidID: http://orcid.org/0000-1111-2222-3333
yubikeyId: 000008001261
passwordFactor: 937b0eb8ae06d881b49e5df4bf12d47c…
bbcDeactivated: FALSE
bbcDeactReason: N/A
bbcHomeInstitute: KI
shadowAccount: 20160101
bbcRegistrar: Ali Gholami
bbcSubjectDN: CN= Ali Gholami nospam@kth.se, O=Kungliga Teknikska
Hogskolan,C=SE,DC=TCS,DC=Terena,DC=org
bbcOnline:False
lastOnline:20141001,18:34
otpSecret:AES128..

group entry
dn: cn=meb00002,ou=Group,dc=biobankcloud,dc=eu
objectClass: top
objectClass: posixGroup
cn: meb00002
gidNumber: 1300002
description: Trusted Researcher

organization entry
dn: ou=KI,ou=Organization,dc=biobankcloud,dc=eu
objectClass: top
objectClass: organizationalUnit

ou: KI
telephoneNumber: +46 8 111 2222
postalAddress: Nobelway 1, 100 44, Solna, Stockhlom, Sweden
resource entry
dn: rn=audit,ou=Resource,dc=biobankcloud,dc=eu
objectClass: top
objectClass: bbcProjectResource
rn: audit

8.1.2 LDAP Server Administration

We implemented our LDAP service using OpenLDAP on an Ubuntu 12.x platform. To install and configure OpenLDAP you should follow the following instructions or use the chef recipes through Vagrant [30].

To create TLS certificates following instructions are required to be performed:

$sudo apt-get install slapd ldap-utils libdb5.1-dev
$sudo apt-get install gnutls-bin ssl-cert
$sudo sh -c "certtool --generate-privkey > /etc/ssl/private/cakey.pem"
$sudo certtool --generate-self-signed --load-privkey /etc/ssl/private/cakey.pem --template /etc/ssl/ca.info --outfile /etc/ssl/certs/cacert.pem
$cd /etc/ldap
$sudo certtool --generate-privkey --bits 1024 --outfile ldapkey.pem
$sudo certtool --generate-certificate --load-privkey ldapkey.pem --load-ca-certificate /etc/ssl/certs/cacert.pem --load-ca-privkey /etc/ssl/private/cakey.pem --template /etc/ssl/ldap01.info --outfile ldapcert.pem
$sudo service slapd restart

$cat > certinfo.ldif
dn: cn=config
add: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ldap/ssl/cacert.pem

add: olcTLSCertificateFile
olcTLSCertificateFile: /etc/ldap/ssl/ldapcert.pem

add: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/ldap/ssl/ldapkey.pem

$sudo ldapmodify -Y EXTERNAL -H ldapi:/// -f certinfo.ldif

Check to see if LDAP service is running:
Place bbc.schema in the current directory. Convert the schema to a dynamic configuration.

$slaptest -f test.conf -F ldap/

This will generate a new schema called {4}bbc.ldif.

Replace the below string in the headers:
--
dn: cn=bbc,cn=schema,cn=config

objectClass: olcSchemaConfig
cn: bbc
--

Also remove the lines after “structuralObjectClass: olcSchemaConfig” at the bottom.

$sudo ldapadd -Y EXTERNAL -H ldapi:/// -f cn\=\{4\}bbc.ldif

To check if schema is added:

$sudo ldapsearch -Q -LLL -Y EXTERNAL -H ldapi:/// -b cn=schema,cn=config
$sudo ldapadd -x -D cn=admin,dc=biobankcloud,dc=eu -w biobankcloud -f bbc_structure.ldif
$sudo ldapadd -x -D cn=admin,dc=biobankcloud,dc=eu -w biobankcloud -f bbc_orgs.ldif
$sudo ldapadd -x -D cn=admin,dc=biobankcloud,dc=eu -w biobankcloud -f bbc_group.ldif
$sudo ldapadd -x -D cn=admin,dc=biobankcloud,dc=eu -w biobankcloud -f bbc_resource.ldif
$sudo ldapadd -x -D cn=admin,dc=biobankcloud,dc=eu -w biobankcloud -f bbc_people.ldif

8.2 Web GUI User Administration

The user administration provides the functionality to search team members belonging to study samples through the GUI, as shown in Figure 17. The study owner entitles other members access to the data through the “Team” tab.

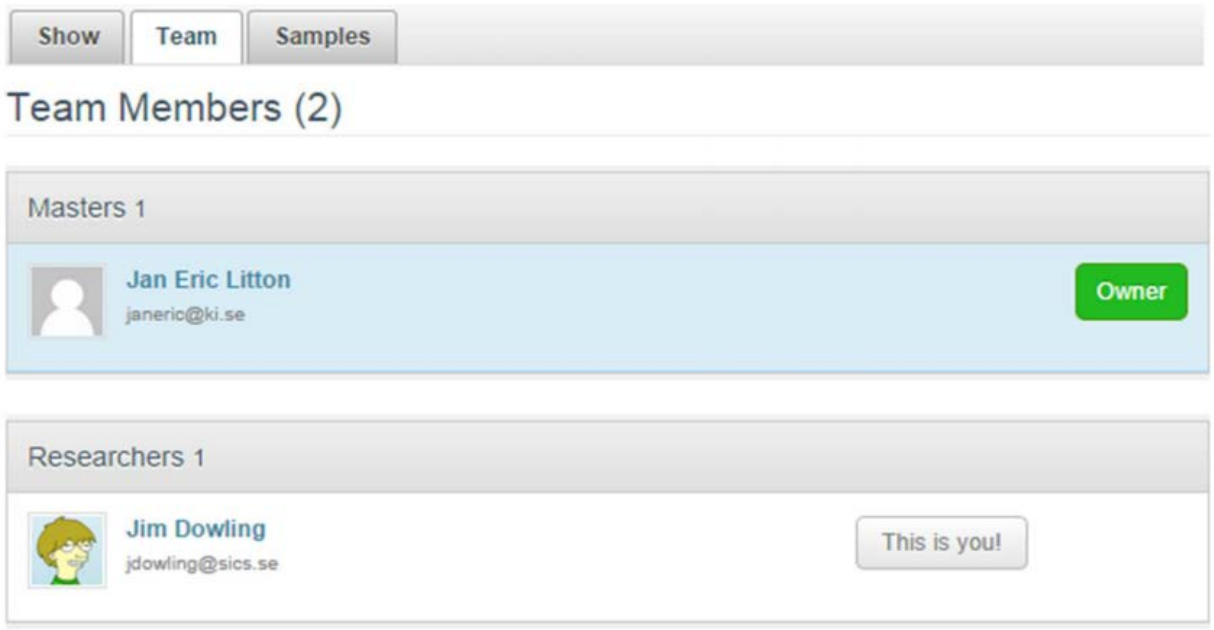

Figure 17, Adding Members to a Study

A study sample owner has enough privileges to entitle different roles to a team that requires access to that specific data set, as shown in Figure 18. The study owner presses the “New Member” button and adds/remove/edit new roles to the members.

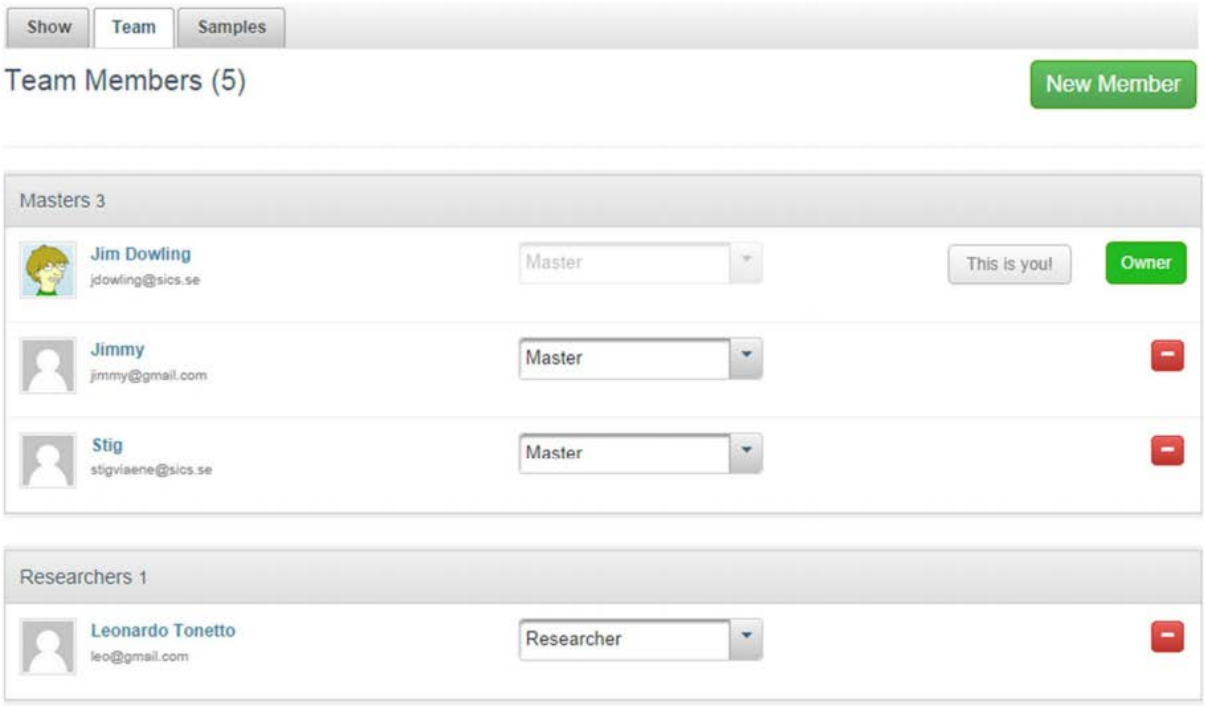

Figure 18, Changing Roles in a Study

9. Discussion

This section describes the functional requirements of the project defined by WP1 and WP6, in addition to the technical requirements of WP2, which required some changes in the design decisions of the deliverables D3.1 state of the art [2] and D3.2 security toolset design [10].

9.1 Functional Requirements

After clarifying the requirements, we realized the platform will not support command-line due to usability. For this purpose, we abandoned developing or integrating extra components for the OpenLDAP management, i.e. user provisioning or account synchronization with the MySQL backend. We decided to change the identification of users using ORCID ID to the POSIX user names. This was decided to see if users can easily use the POSIX usernames otherwise we would implement user identification using email addresses.

9.2 Technical Requirements and Restrictions

The BiobankCloud platform uses Hops infrastructure to provision the services. Hops [13] is implemented using Chef [11] extensively to deploy any service in the BiobankCloud infrastructure. Developing Chef software for each of the security components requires substantial developing efforts that is a limiting factor in WP3. We faced this issue with developing Chef programs when integrating the BiobankCloud Open LDAP service into the Hops infrastructure. WP3 aims to provide security at the platform level and to prioritize our goals on this work package we did some changes in the D3.2 (security toolset design) to fulfil the deliverables objectives.

Moreover, BiobankCloud uses Oracle Glassfish 3.1 to run the platform using Java EE framework. This is a dependency by other components in the project, i.e. study management or job submission to run the workflows. The application server became a limiting factor because of the compatibility of the following items in addition to the development overhead introduced by Chef:

9.2.1 Identification

In D3.2 we decided to use ORCID ID for user identification instead of email. We realized registering new users without an ORCID ID might complicate the registration process. Instead we opted to use the 8-character alphanumeric username and get users feedback on its usability to be finalized in the final toolset. Therefore, if the unique username will not be convenient for users we might reconsider using plain emails as username for identification.

9.2.2 Authentication

Glassfish does not support running two authentication methods in one application. Running X509 certificates method authentication for admin role along with another two-factor authentication method using Yubikey and Mobile authentication. For this purpose we may require to consider integrating supplementary Java EE frameworks such as Apache Shiro [12] to solve the problem. We will decide this problem in the final toolset version in the deliverable D3.4.

9.2.3 Authorization

Argus [14] provides a distributed attribute-based authorization system but is not compatible with the existing GUI and Hops infrastructure. To solve this we use the Java authentication and authorization (JAAS)[9], which support native authorization in Java EE.

9.2.4 Auditing

Due to usability and compatibility of MongoDB[10] with the platform we decided to use MySQL NDB instead to store the log files and audit trails. MongoDB with XDAS [31] were decided to collect the audit rails.

10. Conclusions and Future Work

This deliverable discusses the conceptual architecture of the BiobankCloud and maps the new requirements of the project to the security toolset alpha version. We described implementation of the security toolset alpha version for the BiobankCloud including two-factor authentication using mobile devices and Yubikey tokens in addition to public key certificates. We also presented user administration tools through OpenLDAP and web GUI. Further we discussed the technical limitations of the project that restrict us to use several components that were designed in D3.2 [10].

We have setup a running instance of the security toolset to be tested by the BiobankCloud participants and users. This enables us to improve the alpha version toolset through getting feedback and improving it.

The deliverable security toolset final version (D3.4) due by month 30 will include the final release of the security toolset. In this deliverable we will present the final design, implementation and user guides to use the security toolset.

[9] JAAS, http://docs.oracle.com/javase/8/docs/technotes/guides/security/jaas/tutorials/index.html
[10] MongoDB, http://www.mongodb.org/

References

[1] BiobankCloud- STREP Proposal, FP7 ICT-2011-8

[2] Ali Gholami, Jim Dowling, Roxana Merino Martinez, Salman Niazi, Lora Dimitrova, Jane Reichel, Michael Hummel, Deliverable D3.1, Security State of the Art, WP3, 2013-05

[3] Open Researcher and Contributor ID (ORCID) Structure, http://support.orcid.org/knowledgebase/articles/116780-structure-of-the-orcid-identifier, Accessed: 2014-03

[4] Time-based One-Time Password (TOTP), http://tools.ietf.org/html/rfc6238

[5] The Google Authenticator, https://code.google.com/p/google-authenticator/

[6] HMAC-Based One-Time Password (HOTP), https://tools.ietf.org/html/rfc4226

[7] Sun Java System Web Server 6.1 SP8 Programmer's Guide to Web Applications

[8] Jane Reichel, Roxana Merino Martinez, Jan-Eric Litton, BiobankCloud Deliverable, D1.5 v.01, Regulatory and Ethical Requirements for Biobanking Data Storage and Analysis, 2013-01

[9] Jane Reichel, Roxana Merino Martinez, BiobankCloud Model Data Management Policy (MDMP) and some considerations about the user interface, WP1, 2013-03

[10] Ali Gholami, Jim Dowling, Roxana Merino Martinez, Jane Reichel, Lora Dimitrova, Ulf Leser, Security Toolset Design of the Scalable, Secure Storage BiobankCloud. Deliverable D3.2, 2013-11

[11] Chef, http://docs.chef.io/chef/resources.html, Accessed: 2014-02

[12] Apache Shiro, https://shiro.apache.org/, accessed 2014-12

[13] Hadoop Open Platform (Hop), https://github.com/hopstart, accessed: 2015-01

[14] Argus Authorization Framework, https://twiki.cern.ch/twiki/bin/view/EGEE/AuthorizationFramework, accessed 2014-03

[15] Devaraj Das, Owen O'Malley,Sanjay Radia, Kan Zhang, Adding Security to Apache Hadoop http://hortonworks.com/wp-content/uploads/2011/10/security-design_withCover-1.pdf

[16] Roxana Merino Martinez, Jane Reichel, Jan-Eric Litton, Deliverable, D1.1, Informatics model specification and ethical guidelines for data protection and data sharing, 2013-05

[17] Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified data processing on large clusters. In OSDI, pages 137– 150, 2004

[18] RFC4510, https://tools.ietf.org/html/rfc4510

[19] Jennifer G. Steiner, Clifford Neuman, and Jeffrey I. Schiller. Kerberos: An authentication service for open network systems. In in Usenix Conference Proceedings, pages 191–202, 1988

[20] Hadoop Secure Mode, http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/SecureMode.html, Accessed: 2015-02

[21] Project Rhino, https://github.com/intel-hadoop/project-rhino, Accessed: 2014-06

[22]Owen O'Malley, Kan Zhang, Sanjay Radia, Ram Marti, and Christopher Harrell, Hadoop Security Design, Yahoo!, October 2009

[23] OASIS SAML2 Specification, http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf

[24] RFC 6749, The OAuth 2.0 Authorization Framework, http://tools.ietf.org/html/rfc6749

[25] OpenID specifications, http://openid.net/developers/specs/

[26] Apache Ranger, http://ranger.incubator.apache.org/, accessed: 2015-02

[27] Apache Knox, https://knox.apache.org/, accessed: 2015-02

[28] Apache Sentry, https://blogs.apache.org/sentry/entry/getting_started, Accessed: 2015-02

[29] Hadoop Security Model, http://www.infoq.com/articles/HadoopSecurityModel, Accessed: 2015-02

[30] Vagrant, http://docs.vagrantup.com/v2/provisioning/chef_solo.html, Accessed: 2014-03

[31] OpenXDAS, http://openxdas.sourceforge.net/, accessed: 2013-05