BIOBANKCLOUD
Your PaaS for Biobanking

SEVENTH FRAMEWORK PROGRAMME

Scalable, Secure Storage Biobank

Grant Agreement Number: 317871

BiobankCloud Security:
D3.1, State of the Art

Final

Version:	1.5
Responsible Partner:	Ali Gholami, KTH
Date:	2013-05-30

Project and Deliverable Information Sheet

<table>
<tr><td rowspan="8">Saleable Secure Storage Biobank Project</td><td colspan="2">Project Ref. №: 317871</td></tr>
<tr><td colspan="2">Project Title: Scalable, Secure Storage Biobank</td></tr>
<tr><td colspan="2">Project Web Site: http://www.biobankcloud.eu</td></tr>
<tr><td colspan="2">Deliverable ID: D3.1</td></tr>
<tr><td colspan="2">Deliverable Nature: Report</td></tr>
<tr><td>Deliverable Level:
PU</td><td>Contractual Date of Delivery:
31 /05 /2013</td></tr>
<tr><td></td><td>Actual Date of Delivery:
30 /05 /2013</td></tr>
<tr><td colspan="2">EC Project Officer: Wolfgang Treinen</td></tr>
<tr><td colspan="2">Partner Responsible: Ali Gholami, KTH</td></tr>
<tr><td colspan="2">Contributing Partners: KI & Charité</td></tr>
</table>

* - 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	2012-12-12	Initial version, TOC	Ali Gholami /KTH
0.2	2012-2-26	AAA standards and technologies	Ali Gholami/KTH
0.3	2013-1-29	Regulatory and Ethical Requirements for Biobanking Data Storage and Analysis	Roxana Merino Martinez, Jane Reichel /KI
0.4	2013-3-17	BiobankCloud Model Data Management Policy (MDMP) and some considerations about the user interface	Roxana Merino Martinez, Jane Reichel /KI
0.5	2013-3-19	Gap Analysis, section 6 (Hadoop Security Requirements)	Salman Niazi, Jim Dowling/KTH
0.6	2013-3-25	Related works, threat modelling, evaluation matrix	Ali Gholami/KTH
0.7	2013-4-8	Initial draft	Ali Gholami/KTH
0.8	2013-4-8	General Comments on the draft	Michael Hummel/Charité, Roxana Martinez/KI
0.9	2013-5-7	Final draft, added the utility tree and simplified the evaluation matrix	Ali Gholami/KTH
1.0	2013-5-17	Proof-read of the final draft	Lora Dimitrova, Michael Hummel / Charité
1.1	2013-5-17	Comments to Executive Summary, sections 1, 2.1, and 2.2	Jim Dowling/KTH
1.2	2013-5-22	Revision to comments on versions 1.1	Ali Gholami/KTH
1.3	2013-5-22	Final revision and comments	Lora Dimitrova/ Charité
1.4	2013-5-28	Final revisions and comments	Jim Dowling/KTH
1.5	2013-5-30	Document review	Ali Gholami/KTH

Table of Contents

List of Figures

List of Tables

List of Acronyms and Abbreviation

AAA	**Authentication, Authorization, Auditing**
ABAC	**Attribute-Based Access Control**
ACL	**Access Control List**
AES	**Advanced Encryption Standard**
API	**Application Programming Interface**
AS	**Authentication Service**
AWS	**Amazon Web Services**
BDB	**Berkeley Database**
BWS	**BioID Web Services**
CA	**Certificate Authority**
CM	**Cloud Manager**
CRL	**Certificate Revocation List**
CSA	**Cloud Security Alliance**
CSRF	**Cross-Site Request Forgery**
DAC	**Discretionary Access Control**
DFD	**Data Flow diagram**
DNS	**Domain Name System**
DPaaS	**Data Protection as a Service**
EAP	**Extensible Authentication Protocol**
EC	**European Commission**
ECP	**Enhanced Client or Proxy**
EGEE	**Enabling Grids for E-sciencE**
EK	**Endorsement Key**
ETE	**External Trusted Entity**
EU	**European Union**
FISMA	**Federal Information Security Management Act**
GSM	**Global System for Mobile Communication**
GSS-API	**Generic Security Service Application Program Interface**
HDB	**Hierarchical Database**
HDFS	**Hadoop File System**
HOTP	**HMAC-Based One-Time Password**

HTTP	**Hypertext Transfer Protocol**
HTTPS	**Hypertext Transfer Protocol Secure**
IAM	**Identity and Access Management**
IBE	**Identity-Based Encryption**
IBHMCC	**Identity-Based Hierarchical Model for Cloud Computing**
IBS	**Identity-Based Signature**
IdP	**Identity Provider**
IDS	**Intrusion Detection System**
IETF	**Internet Engineering Task Force**
IP	**Internet Protocol**
JDBC	**Java Database Connectivity**
JSON	**JavaScript Object Notation**
KDC	**Key Distribution Centre**
KI	**Karolinska Institutet**
KTH	**Kungliga Tekniska Högskolan**
LDAP	**Lightweight Directory Access Protocol**
LINDDUN	**Linkability, Identifiability, Non-repudiation, Detectability, Information Disclosure, content Unawareness, and Noncompliance**
MAC	**Mandatory Access Control**
MDB	**In-Memory Database**
MFA	**Multifactor Authentication**
MR	**Map-Reduce**
NDB	**Network Database Technology**
NGS	**Next Generation Sequencing**
OASIS	**Organization for the Advancement of Structured Information Standards**
OATH	**Open AuTHentication**
OAuth	**Open Authorization**
OCRA	**OATH Challenge/Response**
OM	**Object Model**
OP	**OpenID Provider**
OTP	**One-Time Password**
PaaS	**Platform as a Service**
PAP	**Policy Administration Point**
PCI	**Payment Card Industry**
PDP	**Policy Decision Point**
PEP	**Policy Enforcement Point**
PKG	**Private Key Generator**
PKI	**Public Key Infrastructure**
RBAC	**Role-Based Access Control**

REST	**Representational State Transfer**
SAML	**Security Aspersion Markup Language**
SAS	**Statement on Auditing Standards**
SASL	**Simple Authentication and Security Layer**
SCIM	**System for cross-domain identity management**
SDL	**Security Development Lifecycle**
SHA	**Secure Hash Function**
SIM	**Subscriber Identity Module**
SP	**Service Provider**
SPL	**Simplified Policy Language**
SPNEGO	**Simple and Protected GSSAPI Negotiation Mechanism**
SQL	**Structured Query Language**
SSL	**Secure Sockets Layer**
STRID	**Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of privilege**
TCCP	**Trusted Cloud Computing Platform**
TGS	**Ticket Granting Server**
TGT	**Ticket Granting Ticket**
TLS	**Transport Layer Security**
TOTP	**Time-Based One-Time Password**
TPM	**Trusted Platform Module**
TVMM	**Trusted Virtual Machine Monitor**
USB	**Universal Serial Bus**
WAYF	**Where Are You From**
VO	**Virtual Organization**
VOMS	**Virtual Organization Membership Service**
WP	**Work Package**
XACML	**eXtensible Access Control Markup Language**
XDAS	**Distributed Audit Service**
XML	**Extensible Markup Language**
XPath	**XML Path Language**
XSS	**Cross-Site Scripting**

EXECUTIVE SUMMARY

This deliverable D3.1, presents the *state-of-the-art* performed by the WP3 team to identify the existing cloud computing security standards and mechanisms to be used in the project. D3.1 maps these standards and mechanisms to the requirements defined by WP1 *(Regulatory and Ethical Requirements for Biobanking Data Storage and Analysis)* and WP6 *(Integration and Evaluation).* The focus of the deliverable D3.1 is to describe the security requirements for the BiobankCloud, in context of a distributed authentication, authorization and auditing (AAA*)* system where each organization manages its own users and associated privileges. According to the project's description of work, D3.1 shall propose an initial set of tools to be further explored and adapted in the BiobankCloud security framework [97].

D3.1 describes various authentication protocols and standards which can benefit the BiobankCloud such as username/passwords, one-time passwords (OTP), smart cards and physical devices, public key infrastructure (PKI), and biometrics such as face/voice recognition, finger prints. Each of these mechanisms have their own pros and cons that makes it a challenge to choose one or a combination of them, as each mechanism has different advantages, areas of usability, access control, privacy protection, trust management and interoperability.

Another important aspect of BiobankCloud security is discussed in terms of authorization methods which mainly can be classified into role-based access control (RBAC) or attribute-based access control (ABAC). Different mechanisms to address authorization issues such as the integration of an XACML based authorization system or implementing a RBAC using LDAP or other high-throughput solutions are described in section 4.
Auditing requirements are also important to be addressed in the BiobankCloud to ensure secure logging of the AAA events for compliance with the EU Directive 95/46/EC to access and process the sensitive data. Different auditing tools and patterns are discussed such as distributed auditing service (XDAS), Apache Flume and CloudAudit.

Further, this document describes different trust management models and federated identity patterns such as ad hoc, hub-and-spoke and identity network to provide guidelines to design a cross-realm single-sign-on (SSO) solution between different applications and clouds. Different federated identity management solutions and standards such as OpenID, Shibboleth, open authorization (OAuth), security assertion mark-up language (SAML) are also described as options to be used in the BiobankCloud project. Furthermore, we define a set of object model (OM) requirements, described in terms of Hadoop security requirements, to provide guidelines to design a secure object access model that can be integrated with the BiobankCloud security framework.

Finally, D3.1 highlights two trade-off points in the project: strong security vs. usability and interoperability vs. affordability. That an evaluation matrix ranks every component related to AAA based on the attributes defined in the utility tree. We use the evaluation matrix to propose the initial selection of the components that covers the project requirements and constraints. We summarize our recommendations in section 9.

1 Introduction

BiobankCloud is a platform-as-a-service (PaaS) model to store massive amount of molecular data in the cloud and it will provide analytical tools for users belonging to different organizations and institutions [1]. BiobankCloud offers the capability to deploy biobanking applications and tools on the cloud for use by researchers. Biobanks and researchers do not need to worry or deal with the cloud infrastructure administration such as network, servers, operating systems, or storage. However, users' access and protecting the sensitive genomic data in a multi-tenant platform remains a challenging issue that needs to be addressed to ensure compliance with the EU Directive 95/46/EC on personal data protection [2,3,4].

This deliverable describes the security requirements for the BiobankCloud project to be deployed in a private cloud environment. Such security requirements will be addressed through authentication, authorization, auditing (AAA) implemented in an object model (OM) to secure the molecular data. In this document, authentication is identification and it is the process of validating the identity of an end user or a system. Authorization refers to the privileges that the user or system is permitted to do after identity validation through implementing and enforcing policies. Auditing refers to monitoring individuals' activities that access to the molecular data in the BiobankCloud to ensure data integrity and compliance with the EU Directive 95/46/EC [4] on Personal Data Protection.

The rest of this document is structured as follows. Section 2 describes the state of the art in terms of AAA security that can be used in a PaaS model including different standards, protocols, technologies and libraries. Section 3 gives an overview of identity federation concepts and models. Section 4 describes the existing AAA specifications, technologies and protocols that can be used in the BiobankCloud project. In Section 5, related approaches and works that may benefit the project are outlined. Section 6 explains the security requirements related to the object model in context of Hadoop [5]. Section 7 identifies the potential attacks and threats against security and privacy of the BiobankCloud. In Section 8, an initial selection of components is proposed to be further investigated and be deployed as part of the security framework. Finally, Section 9 provides a conclusion of the proposed approaches and it provides guidelines to be continued for the next steps to design the security framework.

2 Review the state-of-the-art

2.1 Identification and Authentication Methods

Identification is the process of a user claiming an identity to a system. There are several identification methods such as user id, account number, email address and MAC/IP address for computing devices. A good identity scheme usually provides *uniqueness* (a unique identifier in different domains), *non-descriptiveness* (not to disclose extra information such as user role or plain names) and *secures issuance* (logging and documentation of identities) [6].

Authentication is verification of the physical user's claimed identity or digital identity of a process or a computer. Users authentication can be categorized within three main categories: authentication by knowledge- e.g., what the user knows such as a password (*memometrics*),

authentication by possession- e.g., what the user has such as a smartcard token (*cognometrics*), and authentication by characteristics- e.g., biometrics such as fingerprints, retinas, iris, voice, face, handwritten *(biometrics)* [7]. These authentication approaches can be combined or used separately, depending on the demanded level of functionality and security.

2.1.1 Password

A password is a simple word or combination of letters and numbers typically between 8-15 characters that is provided by the user upon authentication. Passwords enable a user to claim an identity (usually a user ID or email address) by presenting the knowledge of a secret known only by the owner and the system. In general, passwords should be, as much as possible complex and random. At the same time, the password should be easy to remember by the owner.

Pros: Very common and deployed in many systems**,** convenient for users**,** easy to be implemented**,** no special hardware/software requirements**,** can be changed at the user's request.

Cons: Users often create simple passwords or forget complicated passwords if not used frequently**,** not scalable in multi-server environments**.** In case of encrypted data, loss of password results in the loss of data**,** weak against several attacks such as sniffers, or crackers.

2.1.2 One-Time Password and Physical Devices

One-Time Password (OTP) authentication is a mechanism to decrease the risk of compromising the user credential through generating the OTP tokens by a small device that is similar to a pocket calculator. The idea of OTP is that each session created by the user has a unique credential that is only valid for that session or for a limited duration. Therefore, if an attacker acquires the credentials, they might not be longer valid. There are three OTP approaches:

HMAC-Based One-Time Password (HOTP**):** Tokens are generated based on HMAC-SHA1 flavours using 8-byte counters that increase when a new code is created. The counter must be synchronized between the HOTP generator on the user side and the HOTP validator on the server side. The HOTP generator shares its key with the server [8].

OATH Challenge/Response (OCRA): Tokens that accept a challenge code as input, and generates the response code. OCRA is an extension to HOTP based on random questions. Authentication mode can be one-way just for the client or mutual between both parties in an asynchronous approach to be used in transaction authentication and secure signing [9].

Time-Based One-Time Password (TOTP): Tokens that display a time based password and are updated regularly (default 30 seconds) compared to HOTP where passwords are used for a longer time [10]. As an example, Amazon Web Services (WS) supports TOTP for AWS console logins using an Amazon virtual multifactor authentication (MFA) associated with a single account [11].

The Initiative for Open AuTHentication (OATH) is a collaboration of leading device, platform and application companies working on strong authentication (non-static password) to enable all users and all devices over all networks [12,13]. OATH deals with three major authentication methods such as subscriber identity module (SIM) authentication for global system for mobile communication (GSM) [14], X.509 certificate [15] authentication and OTP based authentication. OATH defines three OTP models in the reference architecture, release 2, and it provides an open standard to implement different OTP approaches using tokens, cell phones, Flash Rom, and other OATH platforms. Moreover, authentication protocols such as Kerberos [16] or other validation and provisioning protocols can be implemented on top of OTP.

Pros: No additional hardware such as reader devices is needed**,** non-static passwords and low administration costs**,** strong authentication against reply attacks**,** wide usage and vendor support such as OATH.

Cons: Complicated Infrastructure setup**,** use of HMAC-SHA1 and OTP seed protection weakness**,** Token life cycle maintenance cost.

2.1.3 Smart Cards

Smart card technology delivers a credit-card sized hardware with an embedded integrated circuit (IC) that must be plugged into a reader device [17]. For example, a flash memory card to store the digital information. A smart card also can be a microprocessor with processing power for encryption capabilities such as public key encryption, signatures, and verification to provide secure identification [18]. Smart cards provide approaches to store a user's certificate and private key.

Pros: Limits the number of logon attempts that locks the smart card**,** lightweight, portable and easy to use**,** tamper-proof identity of users**,** common in the governmental branches with demand on strong authentication.

Cons: The PIN can be forgotten or compromised, requires a reader device, lost smart cards can cause extra costs.

2.1.4 Universal Serial Bus (USB) Tokens

Universal Serial Bus (USB) contains a smart card and a built-in reader chip to provide functionality of both USB token and smart card. USB token delivers two-factor authentication similar to smart cards [7].

Pros: No reading device is required**,** can be easily plugged in to the user devices**,** low deployment costs compared to biometric and smartcard.

Cons: Lost smart cards can cause extra costs**,** must be always carried by the user.

2.1.5 Public Key Infrastructure (PKI) Certificate

Public Key Infrastructure (PKI) provides a scalable secure communication solution in open networks based on an asymmetric pair of keys known as public (shared with all the parties) and private (owned only by the user). PKI provides two main features: a digital signature to sign a document and data encryption between two participants. The usage of the keys is defined through the certificate policy along the validation of the private key and by the public key and certificate revocation list (CRL). A certification authority (CA) distributes such policy [15]. There are several public certificate types that contain additional information such as attributes but the most common is X509 v 3.0 PKI which is built on trust between the issuing CA of the certificate and the public key users.

A X509 certificate policy describes the certificate serial number, user information such as name, company, email, validity period of the certificate, the issuer trusted party who created the certificate and a public key associated with the distinguished name of the user in the certificate.

Pros: Simple authentication without administration efforts of user passwords**,** scalability for the joining organizations and users**,** higher security than basic password authentication**,** capability of creating a proxy certificate for delegation purposes**,** support for SSL/TLS secure communication and single-sign on (SSO).

Cons: Many users don't understand the purpose of certificates**,** certificates must be carried by users and private keys must be kept secure**,** certificate distribution to user and installation can be difficult**,** establishing revocation mechanism in case of compromised private keys.

2.1.6 Biometrics

Biometric authentication methods are based on a measurement of the unique physiological characteristics of a user, such as a fingerprint (to replace password), face recognition, iris code and behavioural characteristics such as voice recognition, handwriting and signature scan [6, 19]. Generally, the identity provider stores the biometric data in encrypted format in a database. Biometric authentication uses pattern matching of the claimed biometric characteristic against a stored template. An accurate biometric authentication balances the equation between a false negative (rejecting authorized user) and a false positive (accepting unauthorized user) rate [20].

Authentication based on fingerprints can be based on embedded sensors or special devices for fingerprint recognition. Some devices such as mice are equipped with embedded sensors which are built in 3D imaging which eliminates the need to buy standalone devices for user authentication.

Voice authentication is another form of biometric authentication which resembles "*what the user is*". This form of biometric is based on voiceprint technology. A microphone captures the user voice and matches it with the backend for successful authentication.

Face recognition is another biometric approach that matched a detected face from the user and validates it against a database of images. There are several vendors supporting biometric

authentication for the cloud such as BioID web services (BWS) and MyBioID (OpenID application based on BWS) as a cloud authentication service for the cloud users based on the standard webcams [21].BioID authentication engine is available for free integration testing that can be used in the BiobankCloud project as proof of concept. OpenCV [22] is another face recognition library that can be used as an authentication engine. However to integrate OpenCV with the cloud services, it requires implementing authentication and validation functions and API.

Pros: No key management issues such as secure key preserving, lost and compromised keys, secure against different threats**,** devices such as webcam and microphone are available in most user devices**,** easy to use by users.

Cons: Voice recognition may not be secure against background noise, sickness, and abnormal conditions**,** not perfect solutions to satisfy the false reject and false accept rate, user privacy concerns, involves cost for support and maintenance and deployment (Fingerprint and iris-scan devices are costly).

2.2 Authorization Techniques

In the BiobankCloud users will login to the platform and based on their role, they will be authorized to perform a set of operations in the system. The BiobankCloud requires access control mechanisms for user provisioning and access control for the applications, processes and data that will be used in the platform. Also the BiobankCloud requires flexible authorization across different platforms within different clouds, because each autonomous domain deploys their own access control mechanism and therefore an authorization system should support multiple policies.

Authorization systems can be mainly classified as centralized or de-centralized. In the centralized approach, one entity is responsible for provisioning access to all the organization and its resources using a consistent and uniform method of controlling access rights. In this model, a single decision authority offers a consistent authorization interface to all the applications in the system. In a decentralized authorization system, service providers authorize the subjects to access the local resources. This approach provides flexibility for different autonomous domains to deploy their own access control list (ACL) mechanisms for local users and services. However, decentralized authorization systems may not always take consistent decisions in distributed environments where users leave and join dynamically and roles are changed frequently.

Generally, access controls are built on user identities or attributes. The well-known identity-based access control mechanisms are discretionary access control (DAC) and mandatory access control (MAC) and role-based access control (RBAC). The second model is called attribute-based access control (ABAC) which deals with the attributes instead of identities. Section 2.2.1 and 2.2.2 describes such access control models in detail.

2.2.1 Identity-Based Access Control (IBAC)

DAC model provides a discretionary access model, because an object owner grants or revokes access to the other subjects (users, programs, and process) in the system. For instance in UNIX, an owner grants different levels of permissions such as read, write, browsing, and execution of a file to other users or groups using ACLs such in AFS [1][23].

Pros: Object owner defines the permissions**,** high flexibility and usability**,** scalable.

Cons: Vulnerable to malicious software such as Trojan horses to change the DAC policies**,** information flow when an object is copied**,** lack of a consistency to enforce local policies in a distributed environment.

The MAC model classifies objects into different levels and assigns each level a label to enforce security clearance for the subjects. Controls are enforced through the central organizational policies and administrators set the object's security level. In this model, object owner cannot grant access to specific users to the data because of high confidentiality of the data such as in military. The object owner only decides what level of permission is sufficient to access the data [24].

Pros: Multilevel security and strong security control over information flow**,** reduced administrative errors or social engineering attacks.

Cons: Administration overhead in dynamic environments with different user requirements.

The RBAC model maps users to roles to access objects in a system based on their organizational positions (permissions) instead of granting access individually. For example, a system administrator is the only one to set and assign the relations. The concept of role is similar to the UNIX group (a collection of users) but a role also assigns permissions to users in addition to extra roles [25]. Also, RBAC provides a separation of duties and least-privilege (giving minimum permissions to accomplish a task) principles to enhance security.

There are four conceptual models of RBAC: RBAC0, RBAC1, RBAC2 and RBAC3. A RBAC0 model provides basic functionality and it consists of users, roles, permissions, operations and objects where roles are flat. RBAC1 uses a hierarchical relation of roles where permissions are inherited. RBAC2 sets constraints to enforce a restrictive rule on the inheritance of permissions for different role configurations. RBAC3 combines functionalities of the other models.

NIST also has standardized the RBAC model, defining it as consisting of two main parts. Reference model to define a common vocabulary of terms to specify the requirements and to set the scope of the RBAC features included in the standard. The second part functional specification defines the requirements of administrative operations to creation and maintenance of the elements and relations. The NIST RBAC model is built on four

[1] Andrew File System (AFS), http://www.openafs.org/

components: Core RBAC, hierarchical RBAC, static separation of duty relations, dynamic separation of duty relations [26].

Pros: Reduced individual administration of users through clearly arranged ACLs**,** when a user changes his role in an organization he automatically gets the privileges for that role**,** the real-world organizational structure can be mapped in the role model.

Cons: The role mapping must be done in the user management system**,** not efficient for dynamic systems where new users' access dynamic services, not scalable across different domains, does not support 1-to-1 mapping.

2.2.2 Attribute-Based Access Control (ABAC)

Attribute-based access control (ABAC*)*, policy-based access control (PBAC), or claim-based access control (CBAC) can be considered as a complementary of the identity-based access controls discussed in section 2.2.1. In ABAC model, when users in one domain change their roles, other domains do not require to update their local systems to be synchronized which is one limitation of RBAC [27].

ABAC introduces the possibility of authorization decisions based on attributes provided by the subjects. Attribute is a separate field that a policy decision point (PDP) consumes against pre-defined values to make authorization decisions. Attributes can be associated with subjects (user, application and process) resources (hosts, objects) and environments (current time, resource usage). XACML is the standard to implement the ABAC model. Open Grid Forum (OGF)[2] has standardized ABAC through SAML to carry attributes of subjects between services for authorization purposes [28, 29].

Pros: Fine-grained permissions than roles, high flexibility in dynamic and distributed environments, capable to define complex policies, compatible with different authentication mechanisms.

Cons: Administration overhead, privileges management.

2.3 Auditing

Auditing is an organized approach for log analysis to ensure that a system is functioning as expected, in order to achieve confidentiality and integrity of sensitive data. A simple model of auditing consists of an audit data controller and an audit data analyzer [29]. The BiobankCloud requires audit and monitoring tools based on logging information of authentication, authorization, access, transfer of the sensitive data to different domains. The purpose of auditing system is to log all events of the AAA framework in a secure fashion for accountability and auditing purposes and providing evidence of compliance for EU Directive 95/46/EC Personal Data Protection requirements [2, 3, 4].

[2] Open Grid Forum (OGF), http://www.gridforum.org/

Logging and log management are important parts of an effective audit process [88]. A log message is a text string generated by the system to show something has happened. Common elements of a log message are: date/time, type of log entry, system, application, successful failure indication, severity and importance, username (in case of users involved) [89]. Log messages can be used for auditing and log analysis to infer meaningful conclusions from the log records such as accountability, tracing illegal usage of the platform, performance, resource usage, fault detection and intrusion detection.

Applications, processes and nodes in the BiobankCloud forward their log messages directly to a log server located in the organization to be stored in the long-term storage. Also deploying a distributed logging system provides the possibility of collecting logs from a networked environment by a log collector to be forwarded to the log server.

The requirements for a log management system should describe the policies to collect, keep and use the logging information. For instance, examples of such policies include enough level of logging, login/logout, resource, intrusion detection systems (IDSs), log retention and log protection against unauthorized access in the storage to ensure integrity of the collected logs.

According to the EU Directive 95/46/EC, auditing and control over personal data is a key requirement. Therefore, BiobankCloud must deploy a reliable auditing system to provide interfaces for internal and external auditing to ensure transparency and integrity of the processed genomic data. In situations when the audit system is not functioning properly, the BiobankCloud platform must be shut down. The platform will register the following events to be used in the audit trails.

2.3.1 Application Events

The BiobankCloud applications should generate log files described by the audit policy to record the applications security related events such as start time, exit time, interaction with the back-ends and external resources (application server, web server, databases) to be used by IDS systems.

2.3.2 User Activities

User activities in the platform such as login/logout, successful or failed attempts to execute applications and processes must be recorded in the audit system to be used by IDS systems.

2.3.3 Object Access

All object access including read, writes, brows, list, modify, delete on the genomic files, folders by users, processes and applications in the BiobankCloud platform should be recorded in the audit system.

2.3.4 User Administration

User provisioning, de-provisioning, assigning or changing the privileges by administrative staff must be registered in the audit system to provide input for the forensics to detect the anomalies of users administration.

3 Identity Federations

Identity federation refers to connecting identity providers (IdPs) and service providers (SPs) via ad hoc, hub and spoke or identity network across organizations for user authentication and authorization to provide single-sign-on (SSO) solutions across different domains. Identity federation enables users to authenticate to an IdPs once and then login to different SPs applications without re-authentication. For example, Eduroam[3] is a well known identity federation enabling students and employees to use the services at thousands of universities worldwide. Identity federation is a desired feature to support in the **BiobankCloud**, so that, users will be able to login with their existing institutional credentials to avoid creating extra username/password or other types of identification and authentication accounts.

3.1 Trust Models

There are three trust models in an identity federation system: pairwise, brokered, and community trust models. In the pairwise model, two IdP directly trust each other based on mutual agreements. The brokered trust model does not establish a direct agreement between different IdPs and SPs, but it offers an intermediate party whom others domains have agreement with. In community trust model, multiple IdPs have common agreements within the community that builds the federation.

3.2 Federated Identity

Identity federation across different applications and domains delivers several benefits to the organizations by reducing user management, enhanced security through enforcing global security and usability of the services for users. Organizations will not require multiple copies of identities for each application and users do not need to have multiple passwords. A central registry can store the identities to be used by all the trusted entities.

3.3 Identity Federation Patterns

There are three identity federation patterns: as ad hoc, hub-and-spoke or an identity network. Ad hoc federation represents a bilateral relationship between a few IdPs and SPs which is established on a one by one basis. In the hub and spoke pattern, a powerful trusted third party known as hub connects several islands of federations. The identity network pattern represents a different approach than ad hoc and hub and spoke by focusing on the technical issues of the identity federation, by providing equal chance to participate in the federation for all partners regardless of their organizational importance [32].

4 AAA Specifications, Technologies and Protocols

This section presents a review of popular AAA solutions with a short review of the pros and cons of each system.

[3] Eduroam, http://www.eduroam.org/

4.1 Kerberos

Kerberos v5 [16] is an authentication protocol based on symmetric key encryption to provide authentication in client/server scenarios in open networks without exposing the shared symmetric key between the principals (client or server instances). Both client and servers mutually authenticate each other without transmission of any password in plain or encrypted form. The Kerberos server sends encrypted tickets with the user shared symmetric key that only can be decrypted by the user knowing the password. A ticket is a token protected by encryption that allows a user to be authenticated without requiring his/her credentials, such as his/her password. Tickets can also be used to identify a user who has been authenticated to use SSO to avoid re-authentication for frequent access. After the authentication phase, encrypted channels will be established to provide secure communication to ensure confidentiality and integrity of the data.

In the Kerberos protocol, the key distribution centre (KDC) is a service composed of the authentication server (AS) and the ticket granting server (TGS). The AS acts as a trusted third-party in charge of user authentication in a realm that users and servers belong. The TGS issues a ticket granting ticket (TGT) during the user first authentication. There are several open source implementations of Kerberos v5 protocol, such as mainly Heimdal v 1.6 [33] and MIT Kerberos [34] versions. Heimdal supports the Kerberos raw API, in addition to PKI, Generic Security Service Application Program Interface (GSS-API) [35]. GSS-API provides an abstraction layer using a standard API for applications to use for the underlying protocols. Heimdal also supports Simple and Protected GSS-API Negotiation Mechanism (SPNEGO) [36], to authenticate client application to a remote server, but neither end knows what authentication protocols the other uses. Simple authentication and security layer (SASL) [37] also implemented in Heimdal as a layer to provide pluggable authentication and data security for the existing application protocols such as lightweight directory access protocol (LDAP) [38]. Heimdal is used in several platforms, including most of the BSD[4] UNIX variants, aside from Apple's Mac OS X[5] and it also supports cross-realm authentication where trust between different realms is required to be established.

Pros: Strong authentication for network based services, cross-realm authentication using cross-realm tickets, delegation through forwardable tickets.

Cons: KDC can become a single point of failure against DoS attacks or insufficient security, brute-force attacks against weak passwords, enabling applications with Kerberos authentication can be a cumbersome task, complicated for ordinary users to setup and use.

4.2 Lightweight Directory Access Protocol (LDAP)

LDAP is a client-server access control protocol for accessing directory services. LDAP defines how entities in a database can be queried and accessed. LDAP defines a simple protocol for updating and searching hierarchical directories over open networks. LDAP v3.0 supports secure communication through transport layer security (TLS v1) to encrypt the

[4] Free BSD, http://www.freebsd.org/

[5] Apple Mac OSX, http://www.apple.com/osx/

communication channel using client and server X.509 certificates by an regular LDAP connection (ldap://:389). Also Netscape secure socket layer (SSL v3) can be used to establish secure connection (ldaps://:636). LDAP can be used to store information about users, including usernames and passwords to be validated by applications.

The LDAP schema specifies objects and attributes and their relation to be stored in the LDAP server in directories for instance "*ou=roles, ou=resource, ou=projects"* to provide scalability in distributed environments. Common attributed of an LDAP entry are: distinguished name (DN) as primary keys, common name (CN), domain component (DC), and organizational unit (OU).

There are different implementations of the LDAP protocol such as OpenLDAP v.24 that is open source and widely popular [39] or active directory (AD) the Microsoft implementation of LDAP that can be utilized through plugins to be integrated with UNIX based systems [40,41] or apache directory [42]. OpenLDAP supports several databases as backend such as Berkeley database (BDB), which is a key/value database, hierarchical Berkeley database (HDB) as default back-ends. Other back-ends such as network database (NDB) to be used with MySQL cluster [43], in-memory database (MDB), LDAP (for proxying requests), SQL (to be used with any relational database) can be used in custom setups [44]. OpenLDAP offers several replication strategies such as multi-master (to allow multiple servers to act as masters), mirror-mode and synchronization-replication mode.

Pros: High availability and resilience topologies, easy to implement and backend with various databases, secure communication using TLS/SSL.

Cons: Suitable for hierarchical structures**,** difficult to implement and deploy with applications, e.g., schemas, replication.

4.3 OpenID

OpenID is an open standard to provide *user-centric* authentication without the need for users to release their credentials such as password or other sensitive credentials to a relying party (RP) who relies the OpenID provider (OP) to offer services. Users create accounts with a OP they prefer and use them for authentication. OpenID authenticates the users dynamically based on the information in the identifier to the related IdP. Identities or subjects identifiers are represented as http or https universal resource locator (URL) or extensible resource identifier (XRI) that can be used in different domains [45].

OpenID delivers federated SSO for web applications in different realms through one identifier provided by the user. Attribute exchange (AX) is an approach to *fetch* or *store* user attributes such as username, email address to RP during authentication phase based on user consent. Provider authentication policy extension (PAPE) is another feature of OpenID to enforce strong authentication by RP through multifactor authentication or anti-phishing authentication methods. OpenID provides a mechanism for simple registration (SREG) to free users from multi-registration in different RPs [29].

OpenID Connect is the next generation of OpenID which operates as a layer on top of OAuth v2.0 to provide passing the delegation access instead of the authentication access to different sites [30].

Pros: Can be easily integrated with the web applications**,** provides user-centric approach**,** trust all users and no need for preconfigured trust**,** commercial support from major cloud vendors and social networks.

Cons: Not interoperable with the networked services**,** no build-in authorization mechanism**,** weak against DNS cache poisoning (http redirect to the OP).

4.4 Open Authorization (OAuth)

Open Authorization (OAuth) v 2.0 [47] is an open standard to provide web applications possibility of authenticating and authorizing users based on their credentials instead of a direct password. For instance, it allows users to share their private resources such as photos, calendar and contact lists stored on one site with another site without releasing their credentials such as username/password.

OAuth specification defines resource server as an entity that protects data. Users are defined as resource owner who grant or deny access to their protected data. The term client refers to the applications that operate on the data, according to the resource owner granted permission acquired by an authorization server, e.g., Google API [46].

Another feature of OAuth is to deliver delegated authorization to grant another application or person to operate on behalf of the data owner. Delegated authorization is done through bearer tokens which is acquired from an authorization server to be sent to other participants to act on behalf of the owner. This token is opaque for clients and can be decoded for the endpoints [46]. OAuth provides interoperability with OpenID Connect for federated Web SSO across different realms.

Pros: Lightweight and compatible with RESTflul Web services**,** strong industry support by vendors such Facebook and Google.

Cons: No support for non-web service authentication**,** weak against DNS cache poisoning.

4.5 The Security Assertion Markup language (SAML)

The Security Assertion Mark-up language (SAML) [48] is an open standard based on XML introduced in 2002 by OASIS for exchanging user authentication and authorization information between IdP and SP. SAML is based on the concept of assertions that can be handed among the client who created the access request, the authentication service, and the access decision point. Assertions are statements about a user that can be passed around between different partners. SAML provides a standard request/response protocol for exchanging XML messages.

SMAL provides "*Portable Trust*" for a verified user in one domain to invoke services in another domain using SSO capabilities within a federation. Also for web services, SAML provides a means by which security assertions about messages and service requesters can be exchanged. SAML is built on XML Schema, XML signature, XML encryption, hypertext transfer protocol (HTTP), and SOAP. SAML IdP and SP establish communication through exchange of SAML metadata.

The SAML specification defines assertions, protocols, bindings and profiles. Assertion (XML) is a claim, statement, or declaration of fact made by some SAML authority about the subjects. SAML protocols define a mechanism to exchange request/response messages and data between different parties. SAML bindings provide a mapping of a SAML protocol and communication protocols such as HTTP and SOAP to send the SAML messages with SOAP. SAML profiles provide message exchange information related to a set of functions to be used by different participants.

There are several implementation of the SAML protocol such as OpenSAML, which is a set of open source C++ and Java libraries. OpenSAML 2, the current version, supports SAML 1.0, 1.1, and 2.0 [49].

Pros: Support for strong authentication and SSO (recommended by CSA)**,** standard message exchange protocol**,** extendable to support different use cases**,** PKI recommended.

Cons: Complexity of SAML 2.0 specification, no dynamic identity provider discovery (usually there is a trust list), major public cloud providers such as AWS and Google App engine don't support SAML.

4.6 Shibboleth

Shibboleth v2.5 is an open-source project that provides federated SSO capabilities to allow sites to make informed authorization decisions for individual access of protected online resources [50]. The Shibboleth software builds SAML 2.0 (section 2.3.3) to deliver federated SSO and attribute exchange framework through authenticating user to their home institutions. A federation in Shibboleth is built through agreement between different organizations where users belong. Users not belonging to a federation are not able to login to SP services.

The Shibboleth identity provider (IdP) provides authentication to users in addition to providing service provider (SP) with several attribute information. Therefore, users don't have to provide manual data every time user logs in to the SP. SP handles the access to a protected resource or application entry point through issuing a SAML authentication request to an identity provider selected by the user- where are you from (WAYF) concept in Shibboleth- to choose the home organization, and processing the authentication response. Shibboleth also supports SAML ECP as an adaption of SSO profile with HTTP and its purpose is to remove the limitations of SSO [51].

For networked-based services that don't rely on web services, Shibboleth has limited support through SAML enhanced client SASL and GSS-API. But none of these mechanisms are used in any production release yet [52].

Pros: Authentication and authorization based on user attributes**,** interoperability with Kerberos, LDAP, JDBC[6], proven solution in higher education and governmental organizations, strong authentication through smartcard/PKI certificates.

Cons: User chooses the IdP and it is not discoverable.

4.7 eXtensible Access Control Markup Language (XACML)

eXtensible Access Control Markup Language (XACML) [53] is a XML based standard which defines a policy language, request/response protocol, and architecture that implements ABAC for access control. However, OASIS also has defined a XACML profile for RBAC [54]. XACML deals with user attributes such as subjects (users), resources (objects which are required to be accessed), actions (operations on resources such as read, write), and the environment (time, location) to create fine-grained policies.

There are several free implementations of XACML such as EGEE Argus, XACMLight [55], XEngine [56]. The Argus authorization framework [57] provides an attribute based authorization decisions software for distributed systems such as web and network, compute and storage. The services are implemented based on the XACML 2.0, and use authorization policies to determine if a user is allowed or denied to perform a certain action on a particular service.

Argus consists of three main components: policy administration point (PAP) to define the policies and store them, policy decision point (PDP) to validate authorization requests against the XACML policies retrieved from the PAP server and policy enforcement point (PEP) server to ensures the integrity and consistency of the authorization requests received from the PEP client. PEP client is a lightweight C/Java library which can be integrated easily with different applications. Argus provides interoperability with Shibboleth and SAML to exchange the XACML attributes through SAML assertions. It also provides the command line capabilities to create human readable policies based simple policy language (SPL) in addition to global banning and unbanning of users.

Pros: Suitable for creating complex authorization policies.

Cons: No standard interaction between PEP, PDP and PAP and client.

4.8 Virtual Organization Membership Service (VOMS)

Virtual organization management service (VOMS) is a centralized system to store authorization information of virtual organizations (VO: a group of collaborating organizations) such as user roles, group hierarchies and capabilities to complement limitations

[6] JDBC Overview, http://www.oracle.com/technetwork/java/overview-141217.html

of ACL [58]. VOMS issues attributes based on X.509 certificate information and SAML attribute assertions [59]. VOMS admin server provides interfaces based on web services to manage and keep track of user roles. VOMS can use relational databases such as MySQL and Oracle as back-end.

Pros: Rich authorization decisions based on roles for VOs, interoperable with SAML and PKI.

Cons: Single point of failure.

4.9 System for Cross-domain Identity Management (SCIM)

System for cross-domain identity management (SCIM) [60] is an industry initiative to define a simple JavaScript object notation (JSON) schema and a standard based on RESTful API for automated user provisioning. SCIM also contains bindings to define transportation of SCIM information with other protocols such as SAML. SCIM core defines attributes such as user, group and resource. SCIM connections can be established through secure channels using TLS/SSL and it provides operation such as GET, PUT, POST, PATCH and DELETE for identity management.

Pros: Lightweight and easy to integrate with cloud application and services, interoperable with SAML.

Cons: Not interoperable with network–based services.

4.10 Flume

Flume is an open source distributed log management streaming mechanism to transfer all log files from different sources to a centralized log server to be analyzed by Hadoop [61]. Flume provides fault-tolerant, scalable and centralized management of the log files based on the ideas of flows. Flume nodes are controller or agents. Each node runs source that accepts the input message from servers or applications and sink to redirect the messages to storage. Channels deliver events of 4kyte size from source to the sink. A Flume stream defines a schema to store the output from the source in the storage where all logs are stored. In distributed settings a master node keeps track of the nodes [62].

Pros: High through put log management system**,** SQL-based language to create complicated queries**,** interoperable with Avro, log4j, syslog, http post with JSON body.

Cons: Difficult to deploy and configure.

4.11 Distributed Audit Service (XDAS)

Distributed Audit Service (XDAS) provides accountability and compliance through an audit service. XDAS specification mainly defines a set of generic events across the system, a portable audit record, API for applications to submit their events to XDAS or read records from an audit trail. XDAS satisfies audit requirements of different scenarios through audit event services, audit service management, audit event management, audit log management,

and audit event enquiry [63]. XDAS defines a standard record format including event, originator, initiator, target, source and data.

In a distributed environment with several platforms, an XDAS agent collects the audit events through the API or direct import and can send them to a centralized event management service. XDAS event consumers will be able to use the XDAS service through audit event analysis service.

OpenXDAS is a complement open source implementation of the XDAS specification including the client-side instrumentation and filtering [64]. There is also the library XDAS4J which is a Java implementation of XDAS [65].

Pros: A consistent auditing framework.

Cons: Difficult to integrate with the existing applications**,** might require development of the API for custom applications.

4.12 CloudAudit/A6 (The Automated Audit, Assertion, Assessment and Assurance API)

CloudAudit/A6 is a group member of Cloud security alliance (CSA) to address audit and compliance in cloud computing for CSPs through a specification for automated audit, assertion, assessment, and assurance API (A6). [66]. CloudAudit goal is to enable users to audit and assess remote infrastructure using platform independent API and namespaces. According to the IETF draft version, CloudAudit provides a set of conventions, standards and API to utilize the HTTP protocol to enable cloud users and external third parties to get automated detailed performance and security statistics about the cloud services. CSA has released a free toolkit for governance, risk and compliance (GRC) consisting of a namespace to clarify how different compliance requirements are defined [67].

Pros: A lightweight audit mechanism for cloud based services.

Cons: No standardization exists yet.

5 Related Work

This section reviews the activities, projects, approaches and standards that have been completed or in progress that impacts the BiobankCloud directly or indirectly, in terms of identity and access management (IAM), federated SSO and processing sensitive.

5.1 Cloud IAM and Federated SSO

There have been numerous amount of work on cloud based authentication and authorization from major cloud PaaS providers such as Google, Microsoft, and salesforce.com. Google App

Engine[7] authenticates users through username/password and it provides federated SSO based OpenID and OAuth [68]. Microsoft Azure[8] offers user authentication based on username/password and public key certificate and a claim-based (claim is a statement that a subject proposes about own or another subject) authorization framework. Salesforce.com has proprietary authentication and provides federated SSO through SAML or delegated authentication. Moreover, several open source projects also have contributed for federated SSO or in the development process such as Moonshot, Contrail and eduGAIN that use similar approaches required for the BiobankCloud federated SSO.

Moonshot [69] is a project lead by JANET, the United Kingdom's research and education network consortium that runs the UK Access Federation. Moonshot will extend the SAML based services to the federated SSO for network-based applications. Moonshot adds SAML, GSS-API, and the extensible authentication protocol (EAP) on top of free RADIUS (similar to the Eduroam approach) [70].

eduGAIN is an infrastructure developed by the GÉANT to simplify the authentication, authorization and data transfer between the federations partners [71]. eduGAIN has a similar approach for data transfer to the BiobankCloud in terms of compliance with the EU law of personal data protection.

Contrail is an EC project that promises to provide a layer for federation private clouds in the health-care sector: "As a rule of thumb: if you want to connect different Clouds together in a federation, the Contrail software is the right solution for your problem. If not the only one available to you" [72].

In 2012, Stihler et al. [73] proposed an integral federated identity management for cloud computing. A trust relationship between the given user and SaaS domains is required so that SaaS user can access the provided application and resources. In PaaS domain, there is an Interceptor that acts as a proxy to accept the user requests and execute it. The interceptor interacts with the secure token service (STS), and requests the security token using the WS-Trust specification.

BaseSpace is a product of Illumina[9] as next-generation sequencing cloud platform for the biologists in collaboration with AWS [74]. The produced genetic data by the NGS machines will be redirected to the amazon data centres and biologists are able to perform specific analysis tasks. Data are secured using AES 256-bit encryption and transferred though SSL channels. Data centres are also compliant with several regulations such as service organization auditing standard No. 70 (SAS70) [91], ISO 2700 [92], payment card industry (PCI) [93], and the Federal Information Security Management Act (FISMA) [94].

Li et al. [75] introduced identity-based authentication for cloud computing using the identity-based hierarchical model for cloud computing (IBHMCC) and related encryption and

[7] Google App Engine, https://developers.google.com/appengine/

[8] Microsoft Windows Azure, http://www.windowsazure.com/en-us/

[9] Illumina, http://www.illumina.com/

signature protocols without using certificates for simplified key management purposes. In the proposed model, identity of node is the DN string from the root node to the current node itself. For instance, identity of entity N *is ID_N = DN_0 || DN_M || DN_N* where *DN_0* is the root entity. IBHMCC is deployed in two modules: Root private ley generator (PKG) with the knowledge of master secret to generate all keys and lower level setup. In this scheme, Root PKG and Lower level setup will be applied iterated on different levels to publicize the public keys. The authors also propose identity-based encryption (IBE) and signature to facilitate secure mutual communication that is common in the cloud. IBE is based on Root PKG and lower level setup algorithms and it is composed of both encryption and decryption while identity-based signature (IBS) is composed of signature to sign the message and verification that is used by other entities to verify the signature.

5.2 Processing Sensitive Data

The BiobankCloud requires computation and processing of the sensitive data. To provide data confidentiality, data owner might encrypt their data prior to storing it on the BiobankCloud and this approach can cause limitations, for the Hadoop cluster to process the data.

Santos et al. [76] extended Terra [77] design where a virtual machine can prove its integrity to the users. The proposed solution is called: trusted cloud computing platform (TCCP) where the whole IaaS is considered as a single system instead of granular hosts in Terra. In TCPM, all nodes run a trusted virtual machine monitor (TVMM) to isolate and protect virtual machines. Users are allowed to access the services by cloud manager (CM). There exists a component known as external trusted entity (ETE) that provides a trust coordinator (TC) service in order to keep track of the trusted VMs (N) in a cluster and attests that the provided services are secure. TCCP guaranties confidentiality and integrity in data and computation and it also enables users to attest to the cloud service provider to ensure whether the services are secure prior to setup their VMs using secure boot and remote attestation. These features are based on the trusted platform module (TPM) chip. The TPM contains an endorsement private key (EK) that uniquely identifies the TPM and some cryptographic functions that cannot be changed.

Data protection as a service (DPaaS) designed by Song et al. [78] offers building blocks of data security for cloud services. This architecture enhances development efforts required to offer data protection while still allowing rapid development and maintenance. In DPaaS key management, access control and logging servers are in a middle tier to facilitate rapid development and user verifications. Each server contains a TPM to ensure secure and attestable execution of the running programs. Sample architecture for data protection as a service illustrates possibility to integrate various technologies such as secure data capsules to encrypted data units or data communication through secure channels. Since the platform enforces all data access, authentications and executable programs, all user and application accesses are maintained in detail in audit logs.

Homomorphic encryption introduces the idea of computing over the encrypted data without knowing the keys on a cloud server. For instance in the BiobankCloud, the data owner for confidentiality reasons may encrypt the data with his public key and store the data in the

cloud. When Hadoop processes the data, there is no need to access the data owner private key to decrypt the data. There are several active research groups who are working on this problem such as Microsoft [79].

Rhino is an initiative started by Intel recently, to design and implement a comprehensive security framework to enhance the current data protection mechanism in Hadoop. It aims at providing a framework for Hadoop key management, authorization, audit and logging. Less is known about the duration of the project and the release of such features to the current Hadoop distribution [80].

5.3 Threat Modelling

Microsoft has proposed a threat modelling for spoofing, tampering, repudiation, information disclosure, denial of service, and elevation of privilege (STRID*)* that is used by several secure software development projects [81]. Secure development lifecycle (SDL) [82] is a newer version of STRIDE to make it easier for non-experts to identify the threats against security and to provide countermeasure against those threats.

More specifically, the CSA highlights nine top threats against cloud computing, including data breaches, data loss, account or service traffic hijacking, insecure interfaces and APIs, denial of service, malicious insiders, and abuse of services [83].

For privacy preserving threat modelling Deng, M. et al., introduced linkability, identifiability, non-repudiation, detectability, information disclosure, content unawareness, and noncompliance (LINDDUN) as a generic methodology for privacy requirement elicitation through mapping the initial data flow diagram (DFD) of the applications' scenarios to corresponding threats [84]. LINDDUN describes the threats in detail using threat tree patterns of common attacks that benefits the software designers.

6 Gap Analysis for Security of the BiobankCloud

This section describes the security requirements and limitations of current approaches to be used in the BiobankCloud project including Hadoop security issues to ensure the confidentiality and integrity of genetic data processed by the Hadoop cluster.

6.1 User Identity

Currently users use their system username to authenticate with the Hadoop cluster. We need to add better support for user identity. One of the main questions here is whether we should attempt to integrate user identity with existing identity services at Biobanks or whether we should specify our own user identity. Another option would be to build a federated identity service, similar to approaches in section 5.1. A requirement we would like to meet, however, is to minimize the effects of changing the authentication system on the existing Hadoop code. Currently, Kerberos is used to identify nodes, not users, and it would be helpful to maintain compatibility with the existing node identification system. Finally, we would solve the

federation problem by providing support for an identification service with widespread support, such as OpenID.

6.2 Data Integrity

Hadoop has support for identifying changes in file blocks through checksums. This provides some support for ensuring the integrity of data that is modified by malicious users. Checksums may, however, not be sufficient to guarantee data integrity of file blocks.

6.3 Data Security

If a data node is compromised then the attacker should not be able to use the security information (token and keys) stored on that node to attack other data nodes. Currently, however, in Hadoop, if an attacker can compromise a single node, it compromises the security of the entire system through the theft of credentials. More information about the existing Hadoop security issues can be found in [85,86].

6.4 Re-identification Risk for Genomic Data

6.4.1 Data Nodes

A compromised data node should not in any way lead to identification of any individual whose genomic data is stored on the machine. Currently the data stored on disk in Hadoop is not encrypted. Genomic data can vary in size from 4 to 150 gigabytes. Data in Hadoop is partitioned into blocks, usually of 64 MB in size, and stored across many data nodes in the system, that is, blocks are striped across nodes in the system.

Currently, there is no link from a block stored on a data node to the file containing the block, i.e., the file containing a user's genomic data. All linkage data, linking files to blocks is stored in the NameNode. We believe that in case of the theft of a data node, the attacker will not be able to easily re-identify the blocks stored on that data node, as there will be no linkage information, linking the block to the file that contains the block. As system size grows, the number of blocks relating to a single user decreases. Therefore, for any reasonably large system, the theft of a data node will give the attacker only, at most, a few blocks of data of any one user's genomic data, and the attacker will not be able to identify the source of the block, i.e., the genome to which it belongs.

Another motivation in favouring the striping of genomic data for security, is that the encryption of data on disk is a big challenge for the Hadoop filesystem (HDFS). Encrypting data in HDFS would require us to provide support for Task Trackers decrypting data to execute MR jobs. This would, in turn, require supporting the delegation of authorization rights to Task Trackers, a complex and challenging problem.

6.4.2 Metadata

Our HDFS system will have multiple metadata servers, consisting of NameNodes and NDB nodes (that store the actual meta-data). NDB nodes are the nodes that make up MySQL Cluster. We need to ensure that NDB nodes avoid any un-authorized access to the meta-data. We propose securing the NDB nodes by locating them on a physically separate network, behind a firewall, with only secure access via authorized clients, see Figure 1. This will require an authorization service.

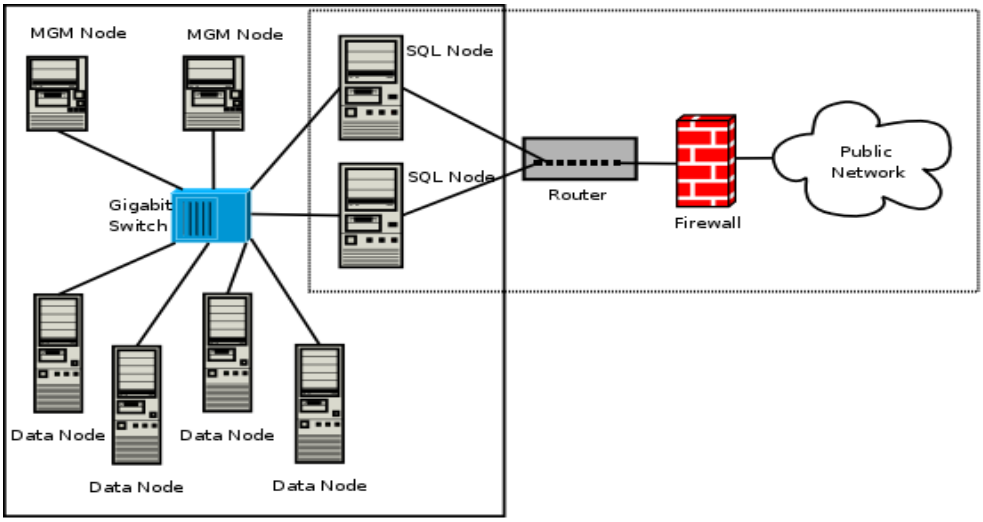

Figure 1, One possible architecture for securing data in MySQL Cluster.

6.5 Authentication

6.5.1 NameNodes

Currently in Hadoop, NameNodes use KDC to authenticate clients and the job tracker. To reduce the authentication traffic of KDC, the NameNode issues delegation token to the clients. Clients use delegation tokens in their further communication with the NameNode. For the BiobankCloud project, this authentication mechanism at the NameNode level would be sufficient, however, Hadoop lacks support for secure user identification. Currently, clients in Hadoop are hosts, not actual users.

6.5.2 DataNodes

When a client wants to read a file, it gets access tokens from the NameNode for all the blocks it wants to read. Each token contains information about the block and the user that is authorized to read it, see Hadoop Security Document for details. Currently the data nodes use this token for authorization only. Data nodes do not perform any user authentication. With the stolen token anyone can read the block from the data node. For the BiobankCloud it is not sufficient. Data nodes need to authenticate all the read and write requests.

6.5.3 Map-Reduce (MR)

For authorization Map-Reduce has the notion of job queues. A user can only submit jobs to certain queues depending on the system configuration. MR has all the inherent authentication weaknesses of HDFS when it comes to actual data block reading. For the BiobankCloud project we would like to improve data access authentication and have more flexible job authorization mechanisms. Support for authorization of file operations needs to be implemented.

6.6 Authorization

Authorization in Hadoop is implemented using Hadoop proprietary tokens. Authorization mechanisms at the NameNode and task tracker level are sufficient for the project. When it comes to authorization at the DataNode level the system has some serious flaws. Anyone who has a data access token for a specific block can read the block from any DataNode that contains the block (typically 3 DataNodes will store copies of each block). Although the data access token contains information about the user who is supposed to be authorized to read the file, the data node does not take any action to actually authenticate the user. The lack of authentication at the DataNode level is mainly a performance issue, as the designers did not want to implement a system that would overload the KDC, as would be the case for large clusters with thousands of nodes, in particular.

6.7 KDC Performance

Current security mechanisms in Hadoop do not put too much strain on the KDC mainly because of Hadoop proprietary delegation tokens. In the BiobankCloud project we would like to implement an authentication mechanism for each block that the client reads or writes.

Some of the biggest Hadoop clusters now have more than 4000 nodes (100 PB). A query/job searching for a particular gene in a study participant's genome file will execute over all of the blocks for the genome file, which are stored across many different data nodes in the system. For a large file, this may result in blocks being processed at all DataNodes in the system. Thus, a job that just searches through a single study participant's would result in an explosion of authorization requests. This would put tremendous strain on KDC. Handling massive numbers of authentication requests is an open research problem. The easiest solution may be to make a high-performance KDC or we would try implementing something like the Hadoop proprietary tokens that do not require constant communication with the KDC for authentication. Another option would be to use MySQL Cluster (NDB) to store authorization lists, as it has high enough performance to handle tens of thousands of authorization requests per second.

7 A Threat Model for the BiobankCloud

In the BiobankCloud PaaS cloud users will not deal with administration of the underlying infrastructure such as operating system through patches and updates, accounts, ports and networks such as switches, routers, firewalls. According to Meier et al. [87] threats against the security of a PaaS model such as Microsoft Azure can be classified into authentication, authorization, auditing/logging, communication, configuration management, cryptography, exception management, input validation, and sensitive data. This section refers to threat as defined by Meier et al. [87] "*A potential occurrence – malicious or otherwise – that can harm an asset*".

7.1 Authentication

Network eavesdropping: An adversary captures network traffic to steal sensitive information such as identity, password and other credentials.

Brute force attacks: An adversary uses a repetitive method to guess identity and/or credentials to get access to the confidential information or administrative level of privileges.

Dictionary attacks: An adversary uses an automated approach to guess identity and/or credentials through the use of common terms in a dictionary file.

Cookie replay attacks: An adversary gains access to an authenticated session through the reuse of a sniffed cookie that server uses to store a value on the victim client.

Credential theft: An adversary gains access to credentials through data theft, for instance, by keystroke logging a compromised computer or phishing or social engineering attacks.

7.2 Authorization

Elevation of privilege: An adversary enters the BiobankCloud as a lower-level user, but is able to gain higher-level access beyond the initially granted rights.

Disclosure of confidential data: An adversary gains access to confidential information such as genetic data or other sensitive information because of authorization failure on a resource or action.

Data tampering: An adversary is able to modify the sensitive data because of authorization failure on a resource or action.

Token stealing: An adversary steals the credentials or token of another principal in order to become authorized to access to the resources. For instance the adversary is able to steal the Kerberos tickets from a Hadoop node and abuse them.

7.3 Auditing and Logging

Disclosure of confidential information: An adversary gathers sensitive information about the genetic data from log files.

Denial of service (DoS): An adversary overloads logs with excessive entries or very large log entries to make audit process incomplete.

Repudiation: An adversary is able to hide his/her exploits of the applications without any trace.

7.4 Communication

Failure to encrypt messages: An adversary is able to capture the traffic contents and read them because of the contents are not encrypted and secure. For instance Hadoop nodes send the

genetic files to each other.

Theft of encryption keys: An adversary is able to decrypt the genetic data because he/she has the encryption keys in a scenario where researchers store encrypted data in the BiobankCloud.

Man-in-the-middle attack: An adversary can read and then modify messages between the client and the server both within the Hadoop cluster and between user and BiobankCloud interactions.

Session replay: An adversary sniffs the exchanged messages and replays them in order to steal a user's session.

Data tampering: An adversary can modify the contents of a message in order to attack the client or the service.

7.5 Configuration Management

Unauthorized access to application repositories: An adversary gains access to configuration files in the application repositories and is able to modify binding settings, etc. for malicious purposes.

Retrieval of clear text configuration secrets: An attacker gains access to configuration files and is able to retrieve sensitive information such as database connection strings or any username/password in the configuration files.

7.6 Cryptography

Encryption cracking: An adversary is able to break the encryption algorithm and gaining access to the encrypted data. This can happen when deploying own made or weak encryption mechanisms.

Loss of decryption keys: An adversary is able to obtain decryption keys and using them to access encrypted genetic data on the cloud. Also, loss of the decryption key by user may result in the loss of the encrypted genetic data, because such encrypted data can never be decrypted without the lost decryption keys.

7.7 Exception Management

Information disclosure: Sensitive system or application details are exposed through exception logs, in case of applications failure.

Denial of service: An adversary uses error conditions to stop your service or place it in an unrecoverable error state. For instance when the BiobankCloud design does not address which part is responsible for which exception.

7.8 Input and Data Validation

Buffer Overflow Attacks: An adversary exploits vulnerability in the applications to acquire administrator privileges or other malicious purposes.

Cross-site scripting (XSS): An adversary injects a client side script into the web pages viewed by other users without checking the contents and sending it back to the users.

Cross-Site Request Forgery (CSRF): An adversary forces the victim's brows to execute forged transactions submitted to a site.

SQL injection: An adversary steals data from a database through SQL injection as input to construct a SQL statement to be executed by server.

XPath injection: XPath injection can result if the input sent to the Web service is used to influence or construct an XPath statement.

XML bomb: An adversary sends an XML bomb with the intent of overloading a Web service's XML parser to cause a DoS attack.

7.9 Sensitive Data

Memory dumping: An adversary is able to read sensitive data out of memory or from local files.

Network eavesdropping: An adversary sniffs the network traffic and learns about unencrypted sensitive data.

Configuration file sniffing: An adversary steals sensitive information, such as connection strings, username/password from configuration files.

Migrating data to third countries: Sensitive genetic data will be transferred to the cloud service providers to non-EU countries.

8 Initial selection of components

8.1 Security Challenges in the BiobankCloud

This section describes the security challenges for the BiobankCloud identity management system in terms of non-functional security requirements adapted from [29] including usability, access control, privacy protection, trust management and interoperability. These requirements are used to build a utility tree to rank the importance of each factor for the BiobankCloud security framework. Finally, we present an evaluation matrix that ranks the discussed standards and components in the earlier sections to facilitate the process of the initial selection of the components to be used in the security framework.

8.1.1 Usability

Usability is one of the most important factors in designing the BiobankCloud security framework, due to the usage of the platform by non-IT experts such as biologists and researchers that are not familiar with complicated access control mechanisms. Also, the frequency of the clients' interaction with the platform highlights the importance of this requirement. Usability can be measured based on the principles of security action (*what a user does to gain access to the system*) and security conclusions (*what users see and infer from the system*). User-friendliness, performance (reasonable physical/mental load of security actions) and affordability (according to the economics of the project such as cost and manpower) are the main attributes of the usability in the project.

8.1.2 Access Control

Different access control mechanisms can be used in the BiobankCloud to provide access to users through enforcing polices. An authorization system that provides a clear definition of different roles such as data controller, data processor, data owner, research, and administrator that are defined in [2]. For instance, RBAC that offers a low administration cost through implementing the role concept or by using an ABAC system with advanced access control features in the project. Therefore, security standards that will be used in the project must be flexible enough to implement either of these models as required.

8.1.3 Privacy Protection

The BiobankCloud security framework must address the requirements of the EU 95/46/EC Directive on Data Protection through communicating the privacy policies between different actors, e.g., data controller, data processor and data owners [2]. Also different security solutions should support implementing various data protection requirements such as consent, lawfulness, transparency, data security (confidentiality and integrity), auditing and control [2,3].

8.1.4 Trust Management

The BiobankCloud requires established trust between different parties and therefore requires the concept of trust to be concrete [3]. For instance, two collaborating institutions that require pre-established agreements to enable their users to access the platform.

8.1.5 Interoperability

The BiobankCloud selected security standards and components should be flexible enough to provide interoperability with existing solutions and also foresee future adoption through a plug-and-play architecture (modifiability of the security mechanism). Portability of the security standards and mechanisms is another attribute that needs to be considered in the security framework design.

8.2 Utility Tree

In order to facilitate the decision making for different architectural scenarios, an utility tree based on the attributes defined in section 8.1 is presented (Figure 2) to communicate the security requirements among all partners within the project. This utility tree also helps to find the architectural trade-off points in the project [95]. The nodes are ranked according to their

overall importance and impact on the security framework: High (H), Medium (M) and Low (L). The second value indicates the risk of achieving such attributes in the project. For instance, a user-friendly system is of high importance and the risk to implement such a system is medium regarding the project and technological constraints.

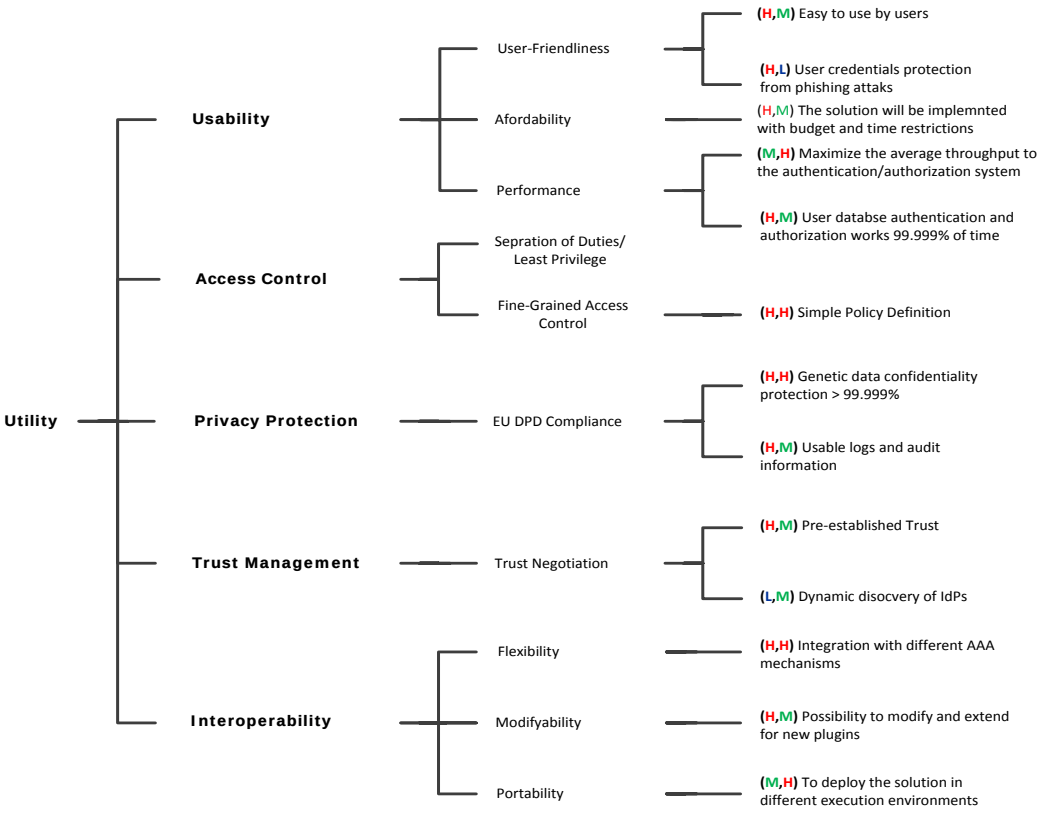

Figure 2, The BiobankCloud Security Utility Tree.

According to this utility tree, there are two main trade-off points in the project: the trade-off between usability and strong security and also trade-off between interoperability and affordability. These trade-off points have been communicated during the project meetings and it is agreed by the project partners to choose an approach that spans the requirements of different groups of users to ensure usable and strong security while considering the interoperability and affordability [96].

8.3 Evaluation Matrix

An evaluation matrix as shown in Table 1, is being used to select the most appropriate components by ranking different options based on the utility criteria defined in section 8.2. The evaluations show there is not a component that fully addresses the security requirements of the project and therefore, it is paramount of importance to choose a combination that fits well in the project.

Table 1, Evaluation Matrix for Initial Selection of Components.

Metric/Solution	Usability	Access Control	Privacy Protection	Trust Management	Interoperability
Username/Password	F	NS	P		P
OTP and Physical Devices	P	NS	F		P
Smart Card	P	NS	F		P
USB Token	P	NS	F		P
PKI	P	NS	F	F	F
Face, Voice Recognition	P	NS	F		F
Kerberos	P	P	P	F	P
LDAP	F	P	P	F	F
OpenID Connect	F	P	P	F	P
OAuth	F	P	F	F	F
SAML	P	P	F	F	F
Shibboleth	P	P	P	F	F
XACML (EGEE Argus)	P	F	P	F	F
VOMS	P	P	P	P	P
SCIM	P		P		P
Flume	P		P		P
XDAS	P		P		P
CloudAudit	P		P		P

* - The evaluation measures are indicated as follows: **F** – Fully, **P** – Partially, **NS** – Not Supported.

8.4 Initial Selection of Components

The preliminary AAA components that can be considered as the building blocks of the security design of the project (D3.1) are mainly the options in section 8.3 which satisfy the trade-off considerations of usability vs. strong security and interoperability vs. affordability. The project must provide username/password authentication for generic users who access the analytical results that are not sensitive information. However, advanced users such as administration staff must use strong authentication (MFA) such as PKI with smart cards or USB Tokens.

To address authorization issues and creating complex authorization policies, EGEE Argus as an ABAC model will be used as authorization service that is also interoperable with Shibboleth. However, Hadoop requires a high throughput authorization service and perhaps a caching mechanism can solve this issue. Shibboleth and SAML also will be used for federated SSO in the project. We will evaluate these solutions but changes may be made during the course of the project.

For auditing requirements, XDAS will be used as a standard to specify the log and audit API and if will be required high volume of logging data, it can be integrated with Apache Flume.

9 Conclusions

BiobankCloud aims to handle a massive amount of molecular data produced by the NGS machines. According to the EU Directive 95/46/EC of Data Protection, genomic data belongs to the sensitive category. Hence, BiobankCloud faces several security problems that must be solved for compliance with the legislations. Such security problem arises for two reasons: the insecure nature of cloud computing such as multi-tenancy, lack of control and the extra privacy requirements introduced by the EU 95/46/EC Directive on Data Protection.

Strong authentication mechanisms along with consistent authorization and auditing will deliver a reliable security framework to provide secure and lawful access across different resources in the BiobankCloud. BiobankCloud security framework should be flexible to support cloud federation for data transfer over different clouds through federated SSO.

Another important feature of the BiobankCloud security framework will be a flexible design to support different authentication, authorization and auditing mechanisms through a plug-and-play philosophy that needs to be taken into account in the design process. For instance, the security framework should provide functionality for integration with new authentication mechanisms such as face recognition or any upcoming technologies.

References

[1] EU FP7 Scalable, Secure Storage Biobank, http://www.biobankcloud.eu/ under Grant No 317871.
[2] 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.
[3] Jane Reichel, Roxana Merino Martinez, BiobankCloud Model Data Management Policy (MDMP) and some considerations about the user interface, WP1, 2013-03.
[4] EU Directive 95/46/EC, http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31995L0046:en:NOT.
[5] Apache Hadoop, http://hadoop.apache.org/.
[6] Harold F. Tipton, Steven Hernandez, Official $(ISC)^2$ Guide to CISSP, CBK, Third Edition, CRC Press 2012.
[7] Lorrie Faith Cranor, Simson Garfinkel, Security and Usability: Designing Secure Systems that People Can Use, O'Reilly Media, 1 edition 2005.
[8] RFC 4226, HOTP: An HMAC-Based One-Time Password Algorithm , http://tools.ietf.org/html/rfc4226.
[9] RFC 6287, OCRA: OATH Challenge-Response Algorithm, http://tools.ietf.org/html/rfc6287.
[10] RFC6238, Time-Based One-Time Password (TOTP), http://tools.ietf.org/html/rfc6238.
[11] Multifactor Authentication, Amazon Web Service, https://aws.amazon.com/mfa/.
[12] Tim Mather, Subra Kumaraswamy, Shahed Latif, Cloud Security and Privacy, An Enterprise Perspective on Risks and Compliance, O'Reilly Media, 2009.
[13] OATH, http://www.openauthentication.org.
[14] Mobile Technologies GSM, http://www.etsi.org/index.php/technologies-clusters/technologies/mobile/gsm.
[15] RFC 2459, Internet X.509 Public Key Infrastructure Certificate and CRL Profile, http://www.ietf.org/rfc/rfc2459.txt.
[16] RFC 4120, The Kerberos Network Authentication Service (V5), http://www.ietf.org/rfc/rfc4120.txt.
[17] Katherine M. Shelfer, J. Drew Procaccino: Smart card evolution. Commun. ACM 45(7): 83-88 (2002).
[18] Types of Smart Cards, http://www.smartcardbasics.com/smart-card-types.html.
[19] Ross J. Anderson, Security Engineering: A Guide to Building Dependable Distributed Systems, Willey, 2008.
[20] Lawrence O'Gorman, Comparing Passwords, Tokens, and Biometrics for User Authentication, Proceedings of the IEEE, Vol. 91, No. 12, 2003.
[21] BioID, http://eu.mybioid.com/.
[22] OpenCV, http://opencv.org/.
[23] Punithasurya K, Jeba Priya S, Analysis of Different Access Control Mechanism in Cloud, International Journal of Applied Information Systems (IJAIS) , Foundation of Computer Science FCS, Volume 4, No.2, 2012.
[24]Mandatory Access Control, http://www.sans.org/reading_room/whitepapers/sysadmin/role-based-access-control-nist-solution_1270.
[25] Ravi Sandhu, and Edward Coyne, and Hal Feinstein, and Charles Youman, Role-Based Access Control Models, Computer, IEEE Computer Society Press, Volume 29, 1996.
[26] Ravi Sandhu, David Ferraiolo, Richard Kuhn, The NIST Model for Role Based Access Control: Toward a Unified Standard, 2000.
[27] Alan H. Karp, Harry Haury, Michael H. Davis, From ABAC to ZBAC: The Evolution of Access Control Model, http://www.hpl.hp.com/techreports/2009/HPL-2009-30.pdf.
[28] Valerio Venturi, Tom Scavo, David Chadwick, Use of SAML to retrieve Authorization Credentials, OGSA Authorization Working Group, 2009.

[29] Elisa Bertino, Kenji Takahashi, Identity Management: Concepts, Technologies, and Systems, Artech House, 2011.
[30] Audit Trails, http://homes.cerias.purdue.edu/~rgk/at.html.
[31] Trust Models Guidelines, https://www.oasis-open.org/committees/download.php/6158/sstc-saml-trustmodels-2.0-draft-01.pdf.
[32] Phillip J. Windley, Digital Identity, O'Reilly, 2005.
[33] Heimdal Kerberos, http://www.h5l.org/.
[34] MIT Kerbeos, http://web.mit.edu/kerberos/.
[35] RFC 2743, Generic Security Service Application Program Interface, http://www.ietf.org/rfc/rfc2743.txt.
[36] RFC4178, The Simple and Protected Generic Security Service Application Program Interface (GSS-API) Negotiation Mechanism, http://tools.ietf.org/html/rfc4178.
[37] RFC 2222, Simple Authentication and Security Layer (SASL), http://www.ietf.org/rfc/rfc2222.txt.
[38] RFC 4510, Lightweight Directory Access Protocol, http://tools.ietf.org/html/rfc4510.
[39] OpenLDAP, http://www.openldap.org/doc/admin24/.
[40] Active Directory Overview, http://technet.microsoft.com/en-us/library/cc758436(v=ws.10).aspx
[41] Windows Security and Directory Services for UNIX Guide v1.0, http://technet.microsoft.com/library/bb463150.aspx).
[42] Apache Directory, http://directory.apache.org/.
[43] MySQL Cluster Carrier Grade Edition: Building a Carrier-Grade Platform for Telecom Data Management, http://www.oracle.com/us/products/mysql/mysql-cluster-opensource-cg-wp-067588.pdf
[44] Matt Butcher, Mastering OpenLDAP: Configuring, Securing and Integrating Directory Services, 2007, ISBN: 1847191029.
[45] OpenID specifications, http://openid.net/developers/specs/.
[46] Ryan Boyd, Getting Started with OAuth 2.0, O'REILLY, 2012.
[47] RFC 6749, The OAuth 2.0 Authorization Framework, http://tools.ietf.org/html/rfc6749.
[48] OASIS SAML2 Specification, http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf.
[49] OpenSAML, https://wiki.shibboleth.net/confluence/display/OpenSAML/Home.
[50] Shibboleth, (http://shibboleth.net/).
[51] SAML V2.0 Enhanced Client or Proxy Profile Version 2.0, https://www.oasis-open.org/committees/download.php/47214/sstc-saml-ecp-v2.0-wd06.pdf.
[52] Enhanced Client or Proxy, https://wiki.shibboleth.net/confluence/display/SHIB2/ECP
[53] http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-cs-01-en.pdf.
[54] XACML v3.0 Core and Hierarchical Role Based Access Control (RBAC) Profile Version 1.0, http://docs.oasis-open.org/xacml/3.0/xacml-3.0-rbac-v1-spec-cd-03-en.html.
[55] XACMLight, http://sourceforge.net/projects/xacmllight/.
[56] XEngine, http://xacmlpdp.sourceforge.net/.
[57] EGEE Argus Authorization Framework, https://twiki.cern.ch/twiki/bin/view/EGEE/AuthorizationFramework..
[58] VOMS: Virtual Organization Membership Service, http://www.globus.org/grid_software/security/voms.php.
[59] EMI VOMS Documentation, https://twiki.cern.ch/twiki/bin/view/EMI/EMIVomsDocumentation.
[60] System for Cross-domain Identity Management, http://www.simplecloud.info/.
[61] Apache Flume, http://flume.apache.org/).
[62] Flume user guide, http://archive.cloudera.com/cdh/3/flume/UserGuide/.
[63] Distributed Audit Service (XDAS), http://hssp-security.wikispaces.com/file/view/Distributed+Audit+Service.pdf.
[64] OpenXDAS, http://openxdas.sourceforge.net/.
[65] XDAS4J, http://xdas4j.codehaus.org/.
[66] CloudAudit 1.0, http://tools.ietf.org/html/draft-hoff-cloudaudit-00.
[67] CloudAudit, https://cloudsecurityalliance.org/research/cloudaudit/.

[68] Google SSO, https://developers.google.com/appengine/articles/openid.
[69] Mooshot Project, https://community.ja.net/groups/moonshot.
[70] Moonshot-enabled Federated Access to Cloud Infrastructure, https://tnc2012.terena.org/core/presentation/14.
[71] eduGAIN, http://www.geant.net/service/eduGAIN.
[72] Contrail, http://contrail-project.eu/en_GB/examples.
[73] Maicon Stihler, Altair Olivo Santin, Arlindo L. Marcon Jr., Joni da Silva Fraga, Integral Federated Identity Management for Cloud Computing, 5th International Conference on New Technologies, Mobility and Security, Istanbul, Turkey, NTMS 2012, May 7-10, 2012.
[74] BaseSpace, https://basespace.illumina.com/home/sequence.
[75] Hongwei Li, Yuanshun Dai, Ling Tian, and Haomiao Yang. Identity-Based Authentication for Cloud Computing. In Proceedings of the 1st International Conference on Cloud Computing. (CloudCom '09), Martin Gilje Jaatun, Gansen Zhao, and Chunming Rong (Eds.). Springer-Verlag, Berlin, Heidelberg, 2009.
[76] Nuno Santos, Krishna P. Gummadi, and Rodrigo Rodrigues, Towards Trusted Cloud Computing, Proceedings of the Workshop On Hot Topics in Cloud Computing (HotCloud), San Diego, CA, June 2009.
[77] Tal Garfinkel, Ben Pfaff , Jim Chow , Mendel Rosenblum , Dan Boneh, Terra: a virtual machine-based platform for trusted computing, ACM Press, 2003.
[78] Dawn Song, Elaine Shi, Ian Fischer, Umesh Shankar, Cloud Data Protection for the Masses, Computer, Vol. 45(1), Jan 2012.
[79] Microsoft CryptoCloud Group, http://research.microsoft.com/en-us/projects/cryptocloud/.
[80] Project Rhino, https://github.com/intel-hadoop/project-rhino/.
[81] The STRIDE Threat Model, http://msdn.microsoft.com/en-us/library/ee823878(v=cs.20).asp.
[82] Secure Development Lifecycle, http://www.microsoft.com/security/sdl/default.aspx.
[83] The Notorious Nine: Cloud Computing Top Threats in 2013, https://cloudsecurityalliance.org/research/top-threats/#_downloads.
[84] M. Deng, K. Wuyts, R. Scandariato, B. Preneel, W. Joosen: A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements, Requirements Engineering, Volume 16 (1) Springer Journals, 2011.
[85] Owen O'Malley, Kan Zhang, Sanjay Radia, Ram Marti, and Christopher Harrell, Hadoop Security Design, http://carfield.com.hk/document/distributed/hadoop-security-design.pdf.
[86] Andrew Becherer, Hadoop Security Design Just Add Kerberos? Really?, http://media.blackhat.com/bh-us-10/whitepapers/Becherer/BlackHat-USA-2010-Becherer-Andrew-Hadoop-Security-wp.pdf.
[87] J.D. Meier, Paul Enfield, Azure Security Notes. Lessons Learned from, Exploring Microsoft Azure and the Cloud Security Space.
[88] NIST Special Publication 800-92, Guide to Computer Security Log Management, http://csrc.nist.gov/publications/nistpubs/800-92/SP800-92.pdf.
[89] Anton A. Chuvakin (Author), Kevin J. Schmidt, Logging and Log Management: The Authoritative Guide to Understanding the Concepts Surrounding Logging and Log Management, Elsevier, 2013.
[90] Ben Halpert, Auditing Cloud Computing: A security and privacy guide, John Willey &Sons, 2011.
[91] Statement on Auditing Standards (SAS) No. 70, http://sas70.com/sas70_overview.html.
[92] ISO 27000, http://www.27000.org/.
[93] Payment Card Industry (PCI), http://www.pcicomplianceguide.org/.
[94] Federal Information Security Management Act (FISMA), http://www.dhs.gov/federal-information-security-management-act-fisma.
[95] Architectural Trade-Off Method (ATAM), http://www.sei.cmu.edu/architecture/tools/evaluate/atam.cfm.

[96] BiobankCloud all-hands meeting, April 22-23, 2013, Charité, Berlin, Germany.
[97] BiobankCloud- STREP Proposal, Call Identifier: FP7 ICT-2011-8.