Security Requirements

Deliverable D 22.2

Document Identification	
Date	27/11/2013
Status	Final Version
Version	1.4

Related SP / WP	SP 3/SP 4/WP 2.2	**Document Reference**	D22.2
Related Deliverable(s)	D22.x	**Dissemination Level**	Public
Lead Participant	Giesecke & Devrient	**Lead Author**	Dr. F.-M. Kamm
Contributors	IFAG, TUD, UNEW, ATOS	**Reviewers**	CA, ULD

This document is issued within the frame and for the purpose of the FutureID project. This project has received funding from the European Unions Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 318424

1. Abstract

The FutureID project develops a comprehensive and privacy-friendly identity management infrastructure for Europe, which integrates existing eID technology and trust infrastructures, emerging federated identity management services and modern credential technologies. In order to obtain acceptance by the users, by service providers and by identity providers it is absolutely essential that the whole FutureID infrastructure ensures a high trust level for its services. This requires a comprehensive security concept for the communication between FutureID components and with external entities and for the handling of user data.

Security is also a major prerequisite for privacy control since any uncontrolled leakage of data may lead to privacy issues. On the other hand, the ability to modify, forge or copy eID data would significantly reduce the trust in FutureID-based identity assurances. Therefore, the whole FutureID system has to be operated as a trusted service with a state-of-the-art security level.

To achieve this, a detailed security concept is required, taking the complexity of the FutureID architecture, the flow of information and the large variety of authentication methods into account. It is the purpose of this document to carefully analyse the security problem, the system boundaries and the resulting threats for the FutureID service and to develop security objectives for the system (chapter 5). In a next step, concrete security requirements for each component will be derived (chapter 6). The methodology follows a simplified version of the Common Criteria approach, yet with a strongly reduced formalism.

While this deliverable concentrates on protection goals like confidentiality, integrity, authenticity and availability independent of the actual content of the data, the privacy requirements (D22.3) will focus more on the handling of person-related data. Specific implementation-related specifications (e.g. which type of algorithm or protocol to be used) can be found in the technical requirement documents for the client (SP 3) and the backend (SP 4).

2. Document Information

2.1 Contributors

Name	Partner
Dr. Frank-Michael Kamm	Giesecke & Devrient GmbH
Nuria Ituarte Aranda	Atos
Dr. Pouyan Sepehrdad	TU Darmstadt
Dr. Detlef Houdeau/Simon Hartmann	Infineon Technologies
Prof. Dr. Thomas Gross	Newcastle University

2.2 History

Version	Date	Author	Changes
0.1	28.01.2013	Frank-Michael Kamm	Initial Version
0.2	13.06.2013	Frank-Michael Kamm	Client requirements added
0.3	19.07.2013	Nuria Ituarte Aranda	AIS requirements added
0.4	19.07.2013	Pouyan Sepehrdad	TUD comments on client integrated
0.5	30.07.2013	D. Houdeau	IFX contribution integrated
0.6	02.08.2013	T. Gross	UNEW Contribution integrated
0.7	08.08.2013	Marit Hansen on behalf of F.-M. Kamm	Adjusting the format of requirements for producing the review version
0.8	19.08.2013	F.-M. Kamm	Corrections of Mapping table
0.9	22.08.2013	F.-M. Kamm	Integration of first reviewer comments
1.0	02.09.2013	T. Gross/ F.-M. Kamm	Integration of AUS mapping and general system requirements
1.1	03.09.2013	M. Drabik/F.-M. Kamm	Integration of further reviewer comments
1.2	03.09.2013	F.-M. Kamm	System requirements mapping
1.3	25.10.2013	F.-M. Kamm	Integrating input from req. harmonization
1.4	27.11.2013	F.-M. Kamm	Renaming of Broker Service

2.3 Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119] .

3. Table of Contents

4. Introduction

The FutureID project develops a comprehensive, flexible, privacy-aware and ubiquitously usable identity management infrastructure for Europe, which integrates existing eID technologies and trust infrastructures, emerging federated identity management services and modern credential technologies to provide a user-centric system for the trustworthy and accountable management of identity claims.

It therefore addresses the currently existing problems and limitations of identity management systems, like the lack of a standardized client, the complex and costly integration of different identity services, privacy threats and the lack of a coherent trust infrastructure.

To solve these problems, FutureID itself has to be a trusted service, offering both the user and the service provider the assurance that authentication using eIDs is reliable, trustworthy and secure while ensuring privacy where needed. Therefore, FutureID has to maintain a high level of security to ensure that neither system components nor critical data are compromised by an attacker.

Taking this overall goal into account, this document specifies the security requirements for FutureID and its components. While this chapter provides a brief introduction and description of the FutureID system and the general security objectives, chapter 5 states the security problem definition and the resulting security objectives. In chapter 6, the concrete security requirements for the various FutureID components are derived.

4.1 System Description

An overview of the FutureID infrastructure is shown in Figure 1. The infrastructure in particular comprises the following components which will be described in this section:

- Client
- Broker Service
- Universal Authentication Service
- Trust Repository
- Application Integration Services

A more complete description of the system components and their functionality can be found in the Description of Work (DoW) of the FutureID project.

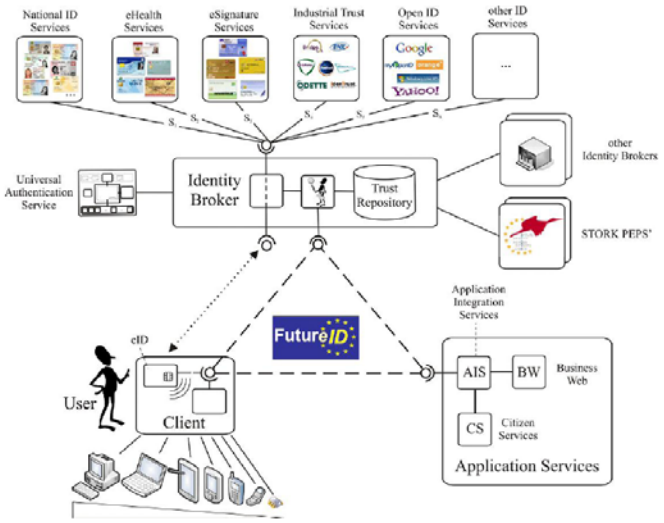

Figure 1: General system architecture of FutureID showing the various system components and their interactions. Source: DoW.

4.1.1 Client

The client acts as an interface between the user (including any user-held eID hardware token) and the FutureID backend infrastructure. To facilitate the broad application of the FutureID technology, the client is designed to support all popular PC platforms and diverse mobile devices including notebooks, tablet PCs, PDAs, smart phones, other mobile phones and even other embedded devices.

4.1.2 Broker Service

The Broker Service will transfer eID information from the client side to the service provider and either works in the Dispatcher Mode, where it only serves as dispatcher and determines an appropriate authentication service, or in the Claims Transformer Mode, where it performs the authentication itself (together with the attached Universal Authentication Service) and then transforms the claims to an appropriate protocol and credential, requested by an external Service Provider. In the Claims Transformer Mode, user attributes could be modified (e.g. filtered).

4.1.3 Universal Authentication Service

The Universal Authentication Service is able to support all authentication protocols implemented by the various authentication tokens deployed across Europe. It makes it possible to support arbitrary authentication protocols by using a generic Execution Environment, which is capable of executing arbitrary protocols, which are described by appropriate Authentication Protocol Specification (APS) files.

As the different authentication protocols are all composed of a rather limited set of basic cryptographic services, the problem of supporting arbitrary authentication protocols is reduced to providing this limited set of basic functionality and providing APS-descriptions for the different authentication protocols.

4.1.4 Trust Repository

The Trust Repository is a database system attached to the Broker Service and provides a comprehensive repository for trusted certificates and services, SAML meta-data for trusted providers and other trust related information.

4.1.5 Application Integration Services

The Application Integration Services allow the communication between the appropriate Federation Service (FS) in the FutureID Infrastructure (see e.g. D21.4) and the Application Services from the service provider.

4.1.6 System Boundaries, interfaces

The system components described above determine the system boundaries of the FutureID system and thus also the boundaries of the security considerations. On the other hand, this system architecture also determines the components which are outside the FutureID boundaries, namely

- the eID token hardware ,
- the client hardware platform and software environment,
- the server and software infrastructure of the service provider who requests eID services from FutureID,
- and the server and software infrastructure of an external identity provider or external Broker Service (e.g. STORK, epSOS, PEPPOL, eSens).

While FutureID services have external interfaces to these components, their security cannot be directly controlled by FutureID components. FutureID services will communicate with these external components and services but can only control the security of a trusted communication channel on the side of the corresponding FutureID communication partner. It has to be assumed that cryptographic algorithms and communication protocols are implemented correctly on these external components and that their systems are in a trustworthy state (e.g. no malware on the server side or client platform).

Internal interfaces exist between the client and the Broker Service, between the client and the Application Integration Services, between the Trust Repository and the Universal Authentication Service and between the Application Integration Service and the federation services.

4.2 General Objectives

FutureID will operate as identity federation system and therefore acts as a third party during the authentication of a user with respect to a specific service. To generate substantial acceptance by users and service providers, it has to run as a trusted service. Therefore, from a security point of view, FutureID has to be able to maintain the required trust and assurance level of an eID despite the large variety of supported eID formats and authentication protocols. FutureID addresses the exchange of authentication and authorization information between the client, the Broker Service, and the Application Integration Service (AIS).

As a consequence, the confidentiality, integrity, and authenticity of eID information and meta-information like the assurance level of a specific eID have to be ensured throughout the whole system and along the external and internal interfaces.

5. Security Objectives

5.1 Assets

The following assets are part of the FutureID system and have to be protected to allow a secure operation and a trustworthy eID federation:

Asset	Description	Protection Goal
eID information	Set of user-related credentials forming the actual eID. This information can be stored in a secure hardware device (e.g. smart card), on a server or mobile device or may be provided by the user input.	Confidentiality, integrity, authenticity, availability.
Passwords/access codes	This asset comprises all passwords or access codes and PINs which are required to authenticate to a separate eID device or to log on to the FutureID infrastructure.	Confidentiality
Private keys	Cryptographic keys which are used for encryption, signature, authentication and other cryptographic services and which may be stored in a secure hardware token, on a server or on a mobile device.	Confidentiality
Public keys and certificates	Certificates including public keys used to verify if the public keys are valid.	Integrity, authenticity, availability
Authentication and authorization data	Data used to authenticate and authorize a user. This could be data generated by a challenge-response-protocol, a Diffie-Hellman based token, a SAML request or something else.	Confidentiality, integrity, authenticity, availability
Meta-information	Any information beyond the actual eID credential data, describing the type, quality, origin or other further authentication information important for eID federation.	Integrity, availability
Assurance level	A special type of meta-information describing	Integrity, availability

Asset	Description	Protection Goal
	the assurance level of the provided eID information. The assurance level determines which type of services can be accessed with a specific eID type.	
Session log files	Log files stored on the FutureID server containing information about the eID used for log on, the time and type of accessed service.	Confidentiality, integrity, availability

5.2 Entities

The following entities are part of the FutureID system or have to be taken into account for security considerations:

Entity	Description
User	The user is the entity (typically a person) which requests access to a certain service and for this reason provides the eID credentials to FutureID.
Client system administrator	The client system administrator is responsible for setting up and maintaining the computing platform on which the FutureID client is running. This can be a PC/Laptop or a mobile device. Especially in private installations, the system administrator may be the same entity as the user. For professional installations however (e.g. in a corporate environment), these roles will be occupied by different persons.
FutureID system administrator	This entity is responsible for setting up and maintaining the backend server infrastructure of FutureID, i.e. the Broker Service, the Universal Authentication Service and the Trust Repository. Depending on the actual installation, this role may be distributed over several persons.
Service provider	The service provider operates the service to which the user wants to obtain access and it is the entity to which FutureID will federate the identity credentials or any derived identity information of the user.
Attacker	An attacker is an adversary which is conducting an adverse action in an unauthorized way (see next section). The attacker can be a local attacker on either the client side or the server side or it can be a remote attacker. A local attacker on the client side may have local access to the client device

Entity	Description
	and/or the eID hardware token, while a local attacker on the server side may have direct local access to the server infrastructure and the storage facilities. In both cases the access will be possible via physical interfaces as well as logical interfaces. A remote attacker will be located outside the FutureID infrastructure and has only access via a logical interface.
Identity Provider	Institution or organization that has issued the identity upon request of or registration by the user. The Identity Provider may also be the issuer of the eID token or other ID credentials that the User uses for logon to FutureID.
Certificate Authority	Institution or organization that issues and revokes certificates for cryptographic keys.

5.3 Adversaries, adverse actions

This section will describe the possible type of attackers and the resulting adverse action in a more detailed way. These considerations will help to state the threats, policies and assumptions which are defined in the next sections.

Adversary	Description
Adverse user	This type of adversary is a malicious user planning to obtain unauthorized access to a service, to modify or forge ID data or to abuse FutureID functionality for any unauthorized or illegal action. An example could be a user trying to obtain access to an age-restricted service by modifying eID credential data.
Client side malware	Malware running on the client device could conduct adverse actions and try to obtain or modify eID data. It could also initiate a sign-on procedure without user consent and could therefore obtain access to user-related services without user permission. eID data could be skimmed on the client side and could be used for unauthorized purposes. In addition, the user behaviour (i.e. type and intensity of services used, type of ID used) could be tracked, leading to a privacy leakage.
Server side malware	Malware running on the server side could conduct similar adverse actions as on the client side, like eavesdropping on ID data, modifying eID or metadata, and obtaining unauthorized access to services. In addition,

Adversary	Description
	unauthorized access to data stored on the server could occur, like data in the Trust Repository and log files. The malware could also try to link different eIDs of the same user and to abuse FutureID functionality, e.g. by circumventing access control policies.
Adverse service provider	An adverse service provider may request eID information from FutureID without being authorized to do so. The service provider could pretend to need certain credential information (e.g. age verification) without actually needing it for a certain service. It could also try to skim eID data without actually offering a service.
Eavesdropper	Beyond eavesdropping on eID information with client or server malware, an eavesdropper could also attack the communication channel between a token and the client, between the client and the FutureID services (e.g. the Broker Service or the Application Integration Service) and between FutureID and a service provider.
Adverse identity provider	An adverse identity provider may request eID information from the client without being authorized to do so. The identity provider could pretend to need certain credential information without actually needing it for authentication.

The adversaries described above could perform any of the following actions or combinations of them:

Adverse Actions	Description
Forge identity	When obtaining access to relevant cryptographic keys used for integrity protection, authentication and access permission, a forged identity could be created that does not exist in reality. In contrast to a modification of parts of an existing identity, a completely new identity would be generated in this case.
Modify eID data	Based on an existing set of eID data, certain credentials may be modified to obtain unauthorized access (e.g. age-based access) or to conceal an existing eID.

Adverse Actions	Description
Modify metadata	By modifying eID metadata an existing eID could be used to obtain unauthorized access to services, e.g. by modifying the assurance level of an eID.
Eavesdropping	By eavesdropping on communication channels eID information could be obtained by an unauthorized entity and could be used for further adverse actions.
Copy eID data	By copying a complete set of eID data, this data may be used for an unauthorized access to a service. If a hardware token is involved, a clone of this token could be generated when all cryptographic information is also accessible.
Delete data	ID information or any permanently or temporarily stored access data could be deleted by an adversary thus preventing access to a service by the user although he may be entitled to obtain access.
Retaining data	ID information or any permanently or temporarily stored access data could be retained although it is required to be deleted (e.g., by law, by user request for deletion or rectification). Adversary may be any involved entity except the user concerned.
Abuse functionality	The functionality of the FutureID infrastructure could be abused by an adversary in case he is able to control important modules of FutureID, like the Universal Authentication Service. In this case an adversary may obtain unauthorized access to services or may be able to skim eID data, passwords or other meta information.
Redirect Data	eID data or metadata could be re-directed by an attacker (e.g. Man-in-the-Middle) and could be used for an authentication which has not been authorized by the user.

5.4 Threats

Based on the definition of assets, entities, adversaries and adverse actions, the following threats can be derived for FutureID:

T.Forged_ID: An adverse user presents a forged ID to the FutureID client based on manipulated data of an existing eID and tries to obtain access to a certain service offered by the

service provider. When FutureID does not detect that the ID has been forged, the user may get unauthorized access to a service. In this case, the integrity of the eID data would be compromised.

T.Eavesdropping: An attacker obtains ID data by eavesdropping on the communication channel between the eID token and the client, between the client and the FutureID server, between internal system components of FutureID or between FutureID and the service provider. Therefore, the security goal is to maintain the confidentiality of the data.

T.Skim: An attacker skims ID data by intercepting the communication on the client side or on the server side of FutureID. The attacker could skim data by conducting a man-in-the-middle attack at one of the internal or external interfaces. In this case, the authenticity of the system components would be compromised.

T.Tracing: An attacker traces eID information by observing the communication between FutureID components. The attacker would be able to unambiguously link an ID to an individual session and could generate profiles about the usage of a certain ID, potentially also across domains. Secret eID information (passwords, keys) does not need to be accessible to the attacker for this kind of attack.

T.Counterfeiting: An attacker counterfeits an ID by producing an unauthorized copy, by extracting eID data and secret cryptographic information or by generating a completely new eID data set. In this case the authenticity of eID data would be compromised.

T.Modification: An attacker modifies metadata like the assurance level data or access rights data. When metadata are modified, an attacker could obtain unauthorized access to a service requiring a high assurance level by providing an eID with a lower assurance level, for example. In this case the integrity of the metadata would be compromised.

T.Abuse: An attacker abuses or modifies functionality of the client or the backend system, for example by circumventing security functionality. In this case the eID federation procedure could lead to incorrect results. Therefore, the overall system functionality would be compromised.

T.Server_leakage: An attacker obtains information stored on the FutureID server (e.g. access data, log files, eID relations…) without authorization. This data could be used to conduct other types of attacks. The leakage of this server data could also lead to privacy issues. Once the attacker has access to the data, he could also modify or delete data, thus threatening the system functionality. In this case the confidentiality and/or integrity of server data would be compromised.

T.Replay An attacker eavesdrops on a successful authentication and replays the same sequence for another authentication. If the communication is encrypted, there is no need for the attacker to decrypt it.

T.MITM An attacker conducts a Man-in-the-Middle attack and is able to eavesdrop on, modify, copy or re-direct eID data, metadata or authentication data or other security-relevant information.

T.Availability An attacker suppresses the availability of system components or system functionality by conducting a Denial of Service (DoS) attack on one or more components or services. Security functionality of the system could thus be made unavailable or could produce false results.

5.5 Organisational security policies

The following organisational policies are assumed for deriving security objectives of FutureID:

OSP.Cl_Standard: The client complies with all relevant communication and interface standards, communication protocols and laws and correctly implements them.

OSP.BE_Standard: The FutureID backend (Broker Service, Application Integration Service, Universial Authentication Service, Trust Repository) complies with all relevant communication and interface standards, communication protocols and laws and correctly implements them.

OSP.Physical: The backend infrastructure of FutureID runs on a site that implements physical site security measures to prevent unauthorized physical access.

5.6 Assumptions

The following assumptions are made concerning FutureID components:

A.Client: The client platform is protected against malware. No malware conducting adverse actions is running in parallel to the FutureID client software.

A.Provider: The service provider implements standards, protocols, cryptography, etc. correctly and is protected against malware and unauthorized access.

A.Personnel: The FutureID backend infrastructure is operated by trustworthy personnel and system administrators.

A.Token: The hardware tokens used for logon to FutureID comply with relevant standards, cryptography requirements, authentication protocols and protection profiles (if available) and correctly implements them. Any eID hardware token is tamper proof.

5.7 Security objectives

Based on the security problem definition above, this section defines the security objectives of the FutureID infrastructure, comprising the Client, the Broker Service, the Application Integration Service, the Trust Repository and the Universal Authentication Service. In addition, the security objectives of the environment are also defined.

5.7.1 FutureID security objectives

The following security objectives shall be ensured by the FutureID infrastructure:

O.Authentication All components of FutureID must use authentication protocols to mutually authenticate. Each communication between the FutureID components, between any hardware token and the FutureID client and between the service provider and the Broker Service shall only take place after a successful mutual authentication.

O.Data_Integrity All components of FutureID must protect the integrity of the eID data, metadata and logfiles during transport and at rest.

O.Confidentiality All components of FutureID must maintain the confidentiality of data during transport and at rest.

O.Tracing All FutureID components must ensure that no tracing of eIDs is possible by unambiguously identifying an ID without the knowledge of secret information or by linking several eIDs of the same user.

O.Access All FutureID components must enforce appropriate access rules such that only authorized persons or instances are allowed to access eID data, metadata or other security relevant data (e.g. cryptographic keys).

O.Sys_Integrity All FutureID components must ensure that the integrity of their software is maintained and no modified software can run in the FutureID system.

O.Cl_Standard: The client must comply with all relevant communication and interface standards, communication protocols and laws and must correctly implement them.

O.IB_Standard: The Broker Service must comply with all relevant communication and interface standards, communication protocols and laws and must correctly implement them.

O.Physical: The backend infrastructure of FutureID must run on a site that implements appropriate physical site security measures to prevent unauthorized physical access to FutureID servers and storage facilities.

O.Replay Every authentication protocol between FutureID internal components and between token and client and between Broker Service and service provider must be resistant against replay attacks.

O.Availability All FutureID components must ensure that the unavailability of their own or other system components does not modify any security functionality.

5.7.2 Objectives of the environment of FutureID

While the security objectives in the previous subsection directly refer to FutureID system components as they were defined in sections 4.1.1 to 4.1.5, some objectives apply to components and entities outside the system boundaries (see section 4.1.6). These components and entities are called the environment of FutureID. The following objectives shall be reached by the environment of FutureID:

OE.Client: The client platform must be protected against malware. No malware conducting adverse actions against FutureID shall run on the client device in parallel to the FutureID client software.

OE.Provider: The service provider and identity provider must implement standards, protocols, cryptography, etc. used for communication with FutureID correctly and must protect their own server infrastructure against malware and unauthorized access.

OE.Personnel: The FutureID backend infrastructure must be operated by trustworthy personnel and system administrators. The entity running the FutureID service must ensure that only trustworthy personnel obtains access to FutureID servers and data.

OE.Token: The hardware tokens used for logon to FutureID must comply with relevant standards, cryptography requirements, authentication protocols and protection profiles (if available) and must correctly implement them. Any eID hardware token used to log on to FutureID must be tamper proof.

5.8 Mapping of objectives

The following table shows a mapping between threats, organisational policies and assumptions with the FutureID objectives and the objectives of the environment.

	O.Authentication	O.Data_Integrity	O.Confidentiality	O.Tracing	O.Access	O.Sys_Integrity	O.Cl_Standard	O.BE_Standard	O.Physical	O.Replay	O.Availability	OE.Client	OE.Provider	OE.Personnel	OE.Token
T.Forged_ID:	x	x	x				x					x			x
T.Eavesdropping:			x		x				x						
T.Skim:	x														
T.Tracing:			x	x	x										
T.Counterfeiting:		x	x		x										x
T.Modification:		x				x									
T.Abuse:					x	x	x	x	x		x				
T.Server_leakage:		x	x		x				x		x				
T.Replay	x									x					x
T.MITM	x	x	x			x			x	x		x		x	x
T.Availability					x	x		x	x		x			x	
OSP.Cl_Standard:							x								
OSP.IB_Standard:								x							
OSP.Physical:									x						
A.Client:												x			
A.Provider:													x		
A.Personnel:														x	
A.Token:															x

Figure 2: Mapping of threats/policies/assumptions vs. security objectives for the client and the client environment.

6. Security Requirements

6.1 Client

6.1.1 Requirements

This section defines the security requirements of the FutureID client. For a better overview, the requirements are grouped according to subtopics.

Identification and Authentication:

SR_CLI_IA_1 The client **must** detect when three unsuccessful authentication events have occurred. When three unsuccessful attempts have been met, the client **must** prevent further authentication for at least ten seconds.

SR_CLI_IA_2 The client **must** maintain the following security attributes of an individual user after an authentication to the FutureID server has occurred and until the session ends: status of authentication (successful/ unsuccessful), trustworthiness level of authentication (see: deliverable D35.1).

SR_CLI_IA_3 All cryptographic secrets generated by the client (e.g. session keys) for FutureID-internal communication **must** initially meet the minimum key lengths of at least 128 bits for AES keys, at least 2048 bits for RSA keys and at least 256 bits for ECC keys. The client **must** be able to detect and enforce that the minimum requirements for a secret are fulfilled. Key lengths **should** be adapted to state-of-the-art recommendations as described in BSI TR-0210202 in its newest version [BSI102] at any time.

SR_CLI_IA_4 The client **should** allow the following actions to be performed before the user is authenticated: selection of authentication method, selection of authentication token (when appropriate). For any other action, the client **must** ensure that the user has successfully been authenticated.

SR_CLI_IA_5 The client **must** prevent reuse of authentication data related to all offered authentication methods.

SR_CLI_IA_6 The client **should** support at least the following user authentication mechanisms: username/password, cryptographic software token and cryptographic hardware token.

SR_CLI_IA_7 The client **must** require a re-authentication of a user in case of following events: session inactivity of more than 5 minutes, loss of connectivity to FutureID server,

device authentication error, data integrity error, failure in establishing a trusted channel, session termination, and session ended.

SR_CLI_IA_8 The client **should** provide only the following feedback to the user, while authentication is in progress: status of authentication (ongoing, completed, or failed).

SR_CLI_IA_9 The client **must** allow no further action until the user has been successfully identified by the FutureID server.

SR_CLI_IA_10 The client **must** associate the following user security attributes with the FutureID session: status of authentication, status of identification, trust level of authentication (see: D35.1). An ID federation event **must** only occur, when the status of authentication and identification are positive and the trust level is available. A change of trust level requires a successful re-authentication with a method required for the specific trust level.

Data Protection:

SR_CLI_DP_01 The client **must** ensure that any security relevant client data, like authentication data, eID data, meta-data, assurance level, log files, etc. is authenticated. The client **must** provide evidence to the FutureID backend, if any of these data have been modified, substituted, re-ordered, or deleted. The client **must** also provide the FutureID backend with the ability to verify evidence of the validity of the integrity assurance.

SR_CLI_DP_02 In case that the FutureID client detects a data integrity error, the client **must** close the current session and request a re-authentication of the user.

SR_CLI_DP_03 The client **must** ensure that all information flow between the client and the FutureID backend, between the client and an eID token and between the client and Application and Integration services occurs only after a successful mutual authentication of the respective endpoints. Information flow **must not** occur without establishing a trusted path (see SR_CLI_TC_01), except for the information required to establish a trusted path.

SR_CLI_DP_04 When eID information imported from an external source (like a hardware token) contains security attributes which are relevant for FutureID, the client **must** enforce that the attributes are unambiguously bound to the eID data and that they are not modified and used only in the originally intended way.

SR_CLI_DP_05 The client **must** enforce that all eID data and meta-data which is transferred between an external token and the client and between the client and the FutureID backend is

encrypted. The used key length of the encryption **must** be in compliance with SR_CLI_IA_3, the used algorithm **must** be in compliance with SR_CLI_CS_04.

SR_CLI_DP_06 The client **must** ensure that any security or privacy relevant data from a previous session is deleted and made unavailable as soon as the session is closed or the communication between the client and the backend or between the client and the Application Integration Service is interrupted.

SR_CLI_DP_07 The client **must** enforce that the transmission of eID data and meta-data is protected from replay errors. The client **must** be able to detect whether replay has occurred. In this case the client **must** terminate the current session and request a re-authentication of the user.

SR_CLI_DP_08 When the client stores any user data or security relevant system data it **must** monitor these data for integrity errors. Upon detection of a data integrity error, the client **must** issue a warning to the user, terminate the current session and request a re-authentication.

Communication:

SR_CLI_CO_01 The client **should** enforce the generation of evidence of origin for received and transmitted eID and meta-data at all times.

Trusted channel:

SR_CLI_TC_01 The client **must** provide a communication channel between itself and another FutureID component or an external eID token that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure.

SR_CLI_TC_02 The client **should** initiate communication via the trusted channel for authentication with an external eID token and for communication with the FutureID backend and the Application Integration Service.

SR_CLI_TC_03 The client **should** permit the FutureID backend to initiate communication via the trusted channel.

Cryptographic Support:

SR_CLI_CS_01 When the client generates cryptographic keys, it **must** ensure that they are in accordance with relevant standards in the most recent version and provide sufficient randomness for distinct applications.

SR_CLI_CS_02 The client **must** ensure that no cryptographic keys can be accessed by the user or by other software outside the FutureID components.

SR_CLI_CS_03 The client **must** delete all temporarily used keys (e.g. session keys) as soon as they are not needed anymore.

SR_CLI_CS_04 For the communication with an eID credential (software or hardware token) the client **must** support the specified cryptographic algorithms and key lengths as defined in the respective specification of each token[1].

SR_CLI_CS_05 For the communication with other FutureID components, the client **must** use the TLS 1.1 protocol (or higher version) [TLS1.1] with minimum cryptographic key sizes as specified in SR_CLI_IA_3.

Protection of Client Security Functionality:

SR_CLI_PS_01 The client **must** verify its own integrity during start up and re-start. In case of an integrity error the client **must** terminate with an according warning message.

SR_CLI_PS_02 The client **should** provide the FutureID backend with the capability to verify the integrity of the client.

SR_CLI_PS_03 The client **should** acknowledge, when requested by another FutureID component, the receipt of an unmodified data transmission.

SR_CLI_PS_04 The client **must** request acknowledgements from other FutureID components for each data transmission on a trusted channel and **must** track the status of each data transmission.

[1] An overview of supported tokens and their cryptographic requirements can be found in D32.1. For signature algorithms, see D33.1.

SR_CLI_PS_05 The client **must** be resistant against run-time attacks which could violate the integrity of client security functionality, like buffer overflows, return-to-libc attacks and return-oriented programming (ROP) attacks. Programming languages and libraries that are susceptible to such attacks must only be used when safe programming is enforced.

Client Access:

SR_CLI_CA_01 The client **must** restrict the maximum number of concurrent authentication sessions that belong to the same user. The client **should** enforce, by default, a limit of one authentication session per user. When more than one concurrent session is allowed, all session communication has to be strictly logically separated.

SR_CLI_CA_02 The client **must** allow user-initiated termination of the user's own FutureID session. The termination **must** be in compliance with SR_CLI_DP_06 and SR_CLI_CS_03.

SR_CLI_CA_03 The client **must** be able to deny session establishment based on an unsuccessful authentication to an eID token or an unsuccessful authentication to the FutureID backend or Application Integration Service.

Security Management:

SR_CLI_SM_01 The client **must** enforce that all security attributes (e.g. authentication status, trust level, etc.) expire after a session has been terminated or has ended. At start up and before a re-authentication, all security attributes have to be initialized.

Security Audit:

SR_CLI_SA_01 The client **must** perform the following actions upon detection of a potential security violation: session termination, residual data deletion in compliance with SR_CLI_DP_06, key destruction in compliance with SR_CLI_CS_03 and security attribute expiration in compliance with SR_CLI_SM_01.

Availability

SR_CLI_AV_01 The Client **must** ensure the operation of the failure state reporting to other FutureID components when software failures occur in one of its modules.

6.1.2 Mapping Requirements vs. Objectives

The following tables provide an overview of the mapping of security requirements versus security objectives. For the sake of readability the mapping is split into two tables.

	O.Authentication	O.Data_Integrity	O.Confidentiality	O.Tracing	O.Access	O.Sys_Integrity	O.CI_Standard	O.BE_Standard	O.Physical	O.Replay
SR_CLI_IA_1	x									
SR_CLI_IA_2	x									
SR_CLI_IA_3		x	x	x						
SR_CLI_IA_4	x				x					
SR_CLI_IA_5	x				x					
SR_CLI_IA_6	x				x					
SR_CLI_IA_7	x				x					
SR_CLI_IA_8	x									
SR_CLI_IA_9	x				x					
SR_CLI_IA_10	x									
SR_CLI_DP_01		x								
SR_CLI_DP_02		x								
SR_CLI_DP_03	x	x	x	x						
SR_CLI_DP_04		x								
SR_CLI_DP_05			x	x						
SR_CLI_DP_06			x	x						
SR_CLI_DP_07										x
SR_CLI_CO_01	x									

Figure 3: Mapping of client security requirements vs. security objectives (part 1).

	O.Authentication	O.Data_Integrity	O.Confidentiality	O.Tracing	O.Access	O.Sys_Integrity	O.Cl_Standard	O.BE_Standard	O.Physical	O.Replay	O.Availability
SR_CLI_TC_01		x	x	x							
SR_CLI_TC_02			x	x							
SR_CLI_TC_03			x	x							
SR_CLI_CS_01		x	x	x			x				
SR_CLI_CS_02		x	x	x							
SR_CLI_CS_03			x	x							
SR_CLI_CS_04			x	x			x				
SR_CLI_CS_05			x	x			x				
SR_CLI_PS_01						x					
SR_CLI_PS_02						x					
SR_CLI_PS_03		x									
SR_CLI_PS_04		x									
SR_CLI_CA_01					x						
SR_CLI_CA_02					x						
SR_CLI_CA_03	x				x						
SR_CLI_SM_01	x		x	x	x						
SR_CLI_SA_01			x	x	x						
SR_CLI_AV_01											x

Figure 4: Mapping of client security requirements vs. security objectives (part 2).

6.2 Broker Service/Trust Repository

6.2.1 Requirements

This section defines the security requirements of the FutureID Broker Service included the Trust Repository. For a better overview, the requirements are grouped according to subtopics.

Identification, Authentication and Authorization

SR_BS_IA_01 The Broker Service **must** use authentication protocols for communicating with the client and the AIS. The components **must** mutually authenticate using these protocols.

SR_BS_IA_01.1 The Broker Service **must** use authentication protocol to the client and AIS such as SAML.

SR_BS_IA_01.1.1 The Broker Service **should** send an answer after the SAML request from the client.

SR_BS_IA_01.1.2 The Broker Service **should** generate and send a session key to the client.

SR_BS_IA_02 The Broker Service **must** use secure transport layer such as TLS 1.1 or higher [TLS1.1] to the client and the AIS in compliance with SR_CLI_IA_3 to support the integrity and authenticity of the data, such as authentication data, eID data, meta-data and the assurance level.

SR_BS_IA_02.1 The Broker Service **should** use a message Authentication Code (MAC) for securing all communication with the client and the AIS.

SR_BS_IA_03 The Broker Service **must** create and send the following statements to the client: authentication statement (the subject was authenticated by a particular means at a particular time), attribute statement (the subject is associated with certain attributes) and authorization decision statement (the subject may be a named resource with a specified action).

SR_BS_IA_04 The Broker Service **must** require a re-authentication with the client in the following cases: session inactivity of more than 5 minutes, loss of the connectivity to the client, device authentication error, data integrity error, failure in establishing a trust channel, session termination and session ended.

SR_BS_IA_05 The Broker Service **must** provide security mechanism to protect the access to sensitive information data from unauthorized access.

SR_BS_IA_05.1 The Broker Service **must** provide security mechanism to protect the access to the Trust Repository data base.

SR_BS_IA_06 The Broker Service **must** provide feedback to the client after the mutual authentication process has finished: status of authentication (ongoing, failed or finished).

SR_BS_IA_07 When the Broker Service transmits the user credentials to the service provider, the transmission **must** be secure.

SR_BS_IA_07.1 The Broker Service **must** use authentication protocol to the service provider such as SAML.

SR_BS_IA_08 The Broker Service **must** select the working mode a) Dispatcher Mode or b) Claims Transformer Mode for the data transmission to the service provider.

SR_BS_IA_09 The Broker Service **should** allow no further action until the user has been successfully identified by the FutureID infrastructure.

SR_BS_IA_10 The Broker Service **must** send the authorization status after routing the eID data to the service provider, e.g. authorization successful or failed.

Data protection

SR_BS_DP_01 The Broker Service **must** ensure that any security relevant data like authentication data, eID data, assurance level, log files are protected against unauthorized modification, substitution, re-ordering, or deletion.

SR_BS_DP_01.1 The Broker Service **should** provide evidence, if any of these data have been modified, substituted or re-ordered.

SR_BS_DP_02 In the case that the Broker Service detects a data integrity error, the Broker Service **must** close the current session and send a request for a re-authentication of the user to the client.

SR_BS_DP_03 The Broker Service **must** ensure that all information flow between the client and the Broker Service, between the AIS and the Broker Service and between the service provider and the Broker Service occur only after a successful mutual authentication of the related end points.

SR_BS_DP_03.1 The Broker Service **must** maintain the confidentiality of the eID data, the authentication data, the meta-data and the assurance level that are sent between the client, the Broker Service, the AIS and the service provider.

SR_BS_DP_03.2 During an authentication event, the Broker Service **must** cross-check the user information with the Trust Repository data base.

SR_BS_DP_04 The Broker Service **must** enforce, that all data, such as authentication data, eID data, meta-data and assurance level data which would be exchanged from the Broker Service to the client, the AIS, and the service provider are encrypted.

SR_BS_DP_05 The Broker Service **must** ensure that any security or privacy relevant person related data from a session are deleted once the session ends.

SR_BS_DP_05.1 The Broker Service **must** ensure that all person-related data are unavailable at the time when the session is closed.

SR_BS_DP_06 The Broker Service **must** enforce that the transmission of eID data and meta-data is protected from replay errors.

SR_BS_DP_06.1 The identity provider **must** be able to detect whether replay has occurred.

Communication

SR_BS_CO_01 The Broker Service **should** enforce the generation of evidence of origin for received and transmitted eID and meta-data at all times.

Trusted channel

SR_BS_TC_01 The Broker Service **must** provide a communication channel between itself and the other FutureID components that provides assured identification of its end points.

SR_BS_TC_01.1 The Broker Service **must** provide a communication channel between itself and the other FutureID components that protect channel data from eavesdropping.

SR_BS_TC_01.2 The Broker Service **must** provide a communication channel between itself and other Broker Service, like STORK, epSOS, PEPPOL and eSenc that provides assured identification of its end points.

SR_BS_TC_02 The Broker Service **must** ensure a communication between itself and the other FutureID components via a trusted channel such as TLS 1.1 or better [TLS1.1] in compliance with SR_CLI_IA_3.

SR_BS_TC_02.1 The Broker Service **must** provide a communication channel between itself and other Broker Service, like STORK, epSOS, PEPPOL and eSenc via trusted channel such as TLS 1.1 or better [TLS1.1] in compliance with SR_CLI_IA_3.

SR_BS_TC_03 The Broker Service **should** ensure communication between itself and the other FutureID components by using secure protocols such as SAML.

Cryptographic Support

SR_BS_CS_01 When the Broker Service creates cryptographic keys for the external communication with client, AIS and service provider, it **must** ensure that they are in accordance with relevant standards in the most recent version and provide sufficient randomness for distinct applications.

SR_BS_CS_01.1 The Broker Service **should** create cryptographic keys for the internal communication with the Trust Repository data base.

SR_BS_CS_02 The Broker Service **must** ensure that no cryptographic keys can be accessed by user or by other software outside the FutureID components.

SR_BS_CS_03 The Broker Service **must** delete all temporarily used keys, such as session key at the time point the session is finished or stopped.

SR_BS_CS_03.1 The Broker Service **must** ensure that the temporarily used keys cannot be recovered.

SR_BS_CS_04 The Broker Service **must** ensure that the communication with other components of FutureID uses the TLS1.1 protocol or higher [TLS1.1] in compliance with SR_CLI_IA_3.

SR_BS_CS_04.1 The Broker Service **should** ensure that the TLS1.1 protocol or higher uses the SSLv2 or SSLv3 encryption in compliance with SR_CLI_IA_3.

Protection of Broker Service Security Functionality

SR_BS_PS_01 The Broker Service **must** verify its own integrity during start-up and re-start or after re-booting.

SR_BS_PS_02 The Broker Service **should** request acknowledgements from the other components of FutureID for each data transmission on a trusted channel.

SR_BS_PS_02.1 The Broker Service **must** track the status of each data transmission.

Broker Service Access

SR_BS_BS_01 The Broker Service **must** restrict a limit of one session per user-ID and **must not** have more than one session per user at the same time. Note: the user can have different roles and based on this circumstance, the user-ID can have different attributes.

SR_BS_BS_02 The Broker Service **must** reject connections with unsuccessful authentication.

SR_BS_BS_03 The Broker Service **must** be implemented on a site with appropriate physical site security measures in order to prevent unauthorized physical access.

SR_BS_BS_03.1 The Broker Service **should** avoid any unnecessary access.

Security Management

SR_BS_SM_1 All attributes, e.g. eID data, authentication data, meta-data and assurance level **must** be deleted after a session has ended.

Security Audit

SR_BS_SA_01 The Broker Service **must** perform the following actions upon detection of a potential security violation: session termination, residual data deletion, key destruction and security attribute expiration.

Availability

SR_BS_AV_01 The Broker Service **must** ensure the operation of the failure state reporting to other FutureID components when software failures occur in one of its modules.

SR_BS_AV_02 The Broker Service **must** assign a priority to each subject in the security functionality. It **must** ensure that access to all sharable resources is mediated on the basis of the assigned priority.

SR_BS_AV_03 The Broker Service **must** enforce maximum quotas for memory space, storage space and CPU load that each authentication session can use during the identity federation procedure.

SR_BS_AV_04 The Broker Service **should** provide sufficient through-put to offer its services under high load, with significant contingency to spare for unexpected events.

SR_BS_AV_05 The modules of the Broker Service **should** provide sufficient resistance against denial-of-service attacks.

6.2.2 Mapping Requirements vs. Objectives

	O.Authentication	O.Data_Integrity	O.Confidentiality	O.Tracing	O.Access	O.Sys_Integrity	O.CI_Standard	O.BE_Standard	O.Physical	O.Replay	O.Availability
SR_BS_IA_01	x										
SR_BS_IA_01.1	x										
SR_BS_IA_01.1.1	x										
SR_BS_IA_01.1.2	x										
SR_BS_IA_02	x		x								
SR_BS_IA_02.1	x	x									
SR_BS_IA_03	x										
SR_BS_IA_04	x										
SR_BS_IA_05					x						
SR_BS_IA_05.1					x						
SR_BS_IA_06	x										
SR_BS_IA_07	x		x								
SR_BS_IA_07.1	x										
SR_BS_IA_08	x										
SR_BS_IA_09	x				x						
SR_BS_IA_10	x										
SR_BS_DP_01		x									
SR_BS_DP_01.1		x									
SR_BS_DP_02		x									
SR_BS_DP_03	x	x	x	x							
SR_BS_DP_03.1	x	x	x	x							
SR_BS_DP_03.2	x	x	x	x							
SR_BS_DP_04			x	x							
SR_BS_DP_05			x	x							
SR_BS_DP_05.1			x	x							
SR_BS_DP_06										x	
SR_BS_DP_06.1										x	

	O.Authentication	O.Data_Integrity	O.Confidentiality	O.Tracing	O.Access	O.Sys_Integrity	O.CI_Standard	O.BE_Standard	O.Physical	O.Replay	O.Availability
SR_BS_CO_01	x										
SR_BS_TC_01		x	x	x							
SR_BS_TC_01.1		x	x	x							
SR_BS_TC_01.2		x	x	x							
SR_BS_TC_02			x	x							
SR_BS_TC_02.1			x	x							
SR_BS_TC_03			x	x							
SR_BS_CS_01		x	x	x				x			
SR_BS_CS_01.1		x	x	x				x			
SR_BS_CS_02		x	x	x							
SR_BS_CS_03			x	x							
SR_BS_CS_03.1			x	x							
SR_BS_CS_04			x	x				x			
SR_BS_CS_04.1			x		x			x			
SR_BS_PS_01						x					
SR_BS_PS_02		x									
SR_BS_PS_02.1		x									
SR_BS_BS_01					x						
SR_BS_BS_02					x						
SR_BS_BS_03		x							x		
SR_BS_BS_03.1					x						
SR_BS_SM_03.1			x	x	x						

	O.Authentication	O.Data_Integrity	O.Confidentiality	O.Tracing	O.Access	O.Sys_Integrity	O.CI_Standard	O.BE_Standard	O.Physical	O.Replay	O.Availability
SR_BS_SA_01			x	x	x						
SR_BS_AV_01											x
SR_BS_AV_02											x
SR_BS_AV_03											x
SR_BS_AV_04											x
SR_BS_AV_05											x

6.3 Universal Authentication Service

6.3.1 Requirements

Identification, Authentication and Authorization

SR_UAS_IA_01 The Universal Authentication Service **must** use mutually authenticated secure channels for communicating with the Broker Service or other components interacting with it.

SR_UAS_IA_02 Should the Universal Authentication Service not be physically collocated with the Broker Service, then the Universal Authentication Service **must** use secure transport layer such as TLS 1.1 or higher [TLS1.1] to the Broker Service to establish a secure channel in compliance with SR_CLI_IA_3 and guarantee confidentiality and integrity of the communication.

SR_UAS_IA_02.1 The Universal Authentication Service **should** only use secure cipher suites and base mutual authentication on client and server certificates.

SR_UAS_IA_03 The Universal Authentication Service **must** require a re-authentication with the Broker Service in the following cases: loss of the connectivity to the client, any authentication error, data integrity error, failure in establishing a trust channel, session termination and session ended.

SR_UAS_IA_04 The Universal Authentication Service **must** provide security mechanism to protect sensitive information from unauthorized access, in particular key material.

SR_UAS_IA_05 When the Universal Authentication Service transmits the user credentials to the Broker Service, the transmission **must** be secure.

Data protection

SR_UAS_DP_01 The Universal Authentication Service **must** ensure that any security relevant data like authentication data, eID data, assurance level, log files are protected against unauthorized modification, substitution, re-ordering, or deletion.

SR_UAS_DP_02 The Universal Authentication Service **should** offer accountability, if any of these data have been modified, substituted or re-ordered.

SR_UAS_DP_03 The Universal Authentication Service **must** maintain confidentiality of non-disclosed data, that is, if pieces of data are not meant to be disclosed the Universal Authentication Service must hold them confidential

SR_UAS_DP_04 In the case that the Universal Authentication Service detects a data integrity error, the Universal Authentication Service **must** close the current session with the Broker Service.

SR_UAS_DP_05 The Universal Authentication Service **must** maintain the confidentiality of the eID data, the authentication data, the meta-data and the assurance level that have been sent.

SR_UAS_DP_06 The Universal Authentication Service **must** enforce that all credentials and key material are held confidential.

SR_UAS_DP_06.1 The Universal Authentication Service **must** ensure that all credentials and key material are securely encrypted at rest.

SR_UAS_DP_06.2 The Universal Authentication Service **must** ensure that all credentials and key material are securely deleted once out of use.

SR_UAS_DP_07 The Universal Authentication Service **must** ensure that any data relevant to security or privacy as well as any person-related data from a session is deleted at closure of the session.

SR_UAS_DP_07.1 The Universal Authentication Service **must** ensure that all person-related pieces of data are unavailable at session-closure time.

SR_UAS_DP_07.2 The Universal Authentication Service **must** ensure that all person-related pieces of data are erased on sessions that failed to close properly.

SR_UAS_DP_07.2 The Universal Authentication Service **must** ensure that all fields for keying material and person-related pieces of data are initialized freshly at session start.

SR_UAS_DP_08 The Universal Authentication Service's communication **must** be protected against replay attacks.

SR_UAS_DP_08.1 The Universal Authentication Service's request and response messages **must** be protected against replay attacks.

SR_UAS_DP_08.2 The Universal Authentication Service **must** be able to detect whether replay has occurred.

SR_UAS_DP_09 The logging of the Universal Authentication Service **must** be in accordance with privacy regulations and requirements.

Data Integrity

SR_UAS_DI_01 The Universal Authentication Service **must** ensure the integrity of eID data transformed in authentication protocols.

SR_UAS_DI_02 The Universal Authentication Service **must** ensure that data transformations are done trustworthy.

SR_UAS_DI_03 The Universal Authentication Service **must** ensure that data included in new security tokens follows the token specification.

Trusted channel

SR_UAS_TC_01 The Universal Authentication Service **must** provide a communication channel between itself and the other FutureID components that provides assured identification of its end points.

SR_UAS_TC_01.1 The Universal Authentication Service **must** provide a communication channel between itself and the Broker Service or other FutureID components that protects channel data from eavesdropping.

SR_UAS_TC_01.2 The Universal Authentication Service **must** provide a communication channel between itself and the Broker Service or other FutureID components that is secure against man-in-the-middle attacks.

SR_UAS_TC_02 Should the Universal Authentication Service not be physically collocated with the Broker Service, then the Universal Authentication Service **must** ensure a secure communication between itself and the Broker Service via a trusted channel such as TLS 1.1 or better [TLS1.1] in compliance with SR_CLI_IA_3.

Cryptographic Support

SR_UAS_CS_01 The Universal Authentication Service **must** generate cryptographic keys in accordance with relevant standards in the most recent version which provide sufficient randomness for distinct applications.

SR_UAS_CS_02 The Universal Authentication Service **should** create cryptographic keys for the external communication with the Broker Service.

SR_UAS_CS_03 The Universal Authentication Service **must** ensure that no secret cryptographic keys can leak outside of its operational environment.

SR_UAS_CS_04 The Universal Authentication Service **must** delete all ephemeral keys, such as session keys, at the time point the session is completed or interrupted.

SR_UAS_CS_04.1 The Universal Authentication Service **must** ensure that ephemeral keys cannot be recovered.

SR_UAS_CS_04.2 The Universal Authentication Service **must** ensure that ephemeral keys are deleted if the service recovers from a failure state.

Protection of Universal Authentication Service Security Functionality

SR_UAS_PS_01 The Universal Authentication Service **must** verify its own integrity during start-up and re-start or after re-booting.

SR_UAS_PS_02 The Universal Authentication Service **must** verify the correct parsing and interpretation of all its APS language specifications.

SR_UAS_PS_03 The Universal Authentication Service **must** check the consistency of all its modules and libraries with its APS language repository, including matching versions.

SR_UAS_PS_04 The Universal Authentication Service **should** check the consistency of specifications in APS language with the general security policies, including key lengths, cryptography parameters, and allowed protocol suites.

SR_UAS_PS_05 The Universal Authentication Service **must** handle exception states, failure modes, errors and faults securely and systematically.

SR_UAS_PS_06 The Universal Authentication Service **should** offer a fail-over mechanism for failures to enable a secure and controlled return into operational state.

SR_UAS_PS_07 The Universal Authentication Service **should** communicate failure states systematically to the Broker Service, where the communication must not leak sensitive information.

SR_UAS_PS_08 The Universal Authentication Service **must** enforce a clean-up after exception states, failure modes, errors and faults that ensures that key material and personal data of the failed session is deleted securely.

SR_UAS_PS_09 The Universal Authentication Service **must** be resistant against run-time attacks which could violate the integrity of Universal Authentication Service security functionality, like buffer overflow or XML format attacks. Programming languages and libraries that are susceptible to such attacks must only be used when safe programming is enforced.

Universal Authentication Service Access

SR_UAS_BS_01 The Universal Authentication Service **must** reject connections with unsuccessful authentication.

SR_UAS_BS_02 The Universal Authentication Service **must** be implemented on a site with appropriate physical site security measures in order to prevent unauthorized physical access.

SR_UAS_BS_03 The Universal Authentication Service **should** be shielded from unauthorized access apart from the Broker Service by firewall.

Availability

SR_UAS_AV_01 The Universal Authentication Service **must** ensure the operation of the failure state reporting to other FutureID components when software failures occur in one of its modules.

SR_UAS_AV_02 The Universal Authentication Service **must** assign a priority to each subject in the security functionality. It **must** ensure that access to all sharable resources is mediated on the basis of the assigned priority.

SR_UAS_AV_03 The Universal Authentication Service **must** enforce maximum quotas for memory space, storage space and CPU load that each authentication session can use during the identity federation procedure.

SR_UAS_AV_04 The Universal Authentication Service **should** provide sufficient through-put to offer its services under high load, with significant contingency to spare for unexpected events.

SR_UAS_AV_05 The modules of the Universal Authentication Service **should** provide sufficient resistance against denial-of-service attacks.

Security Management

SR_UAS_SM_03 All attributes, e.g. eID data, authentication data, meta-data and assurance level **must** be deleted after a session has been ended.

SR_UAS_SM_03.1 All ephemeral key material **must** be deleted after a session has been completed.

Security Violations

SR_UAS_SV_01 The Universal Authentication Service **must** perform the following actions upon detection of a potential security violation: session termination, residual data deletion and key destruction.

6.3.2 Mapping Requirements vs. Objectives

	O.Authentication	O.Data_Integrity	O.Confidentiality	O.Tracing	O.Access	O.Sys_Integrity	O.CI_Standard	O.BE_Standard	O.Physical	O.Replay	O.Availability
SR_UAS_IA_01	X										
SR_UAS_IA_02	X										
SR_UAS_IA_03	X	X			X					X	
SR_UAS_IA_04	X		X		X						
SR_UAS_IA_05	X				X						
SR_UAS_DP_01		X									
SR_UAS_DP_02	X	X									
SR_UAS_DP_03	X		X	X							
SR_UAS_DP_04		X									
SR_UAS_DP_05			X	X							
SR_UAS_DP_06			X	X							
SR_UAS_DP_07			X	X							
SR_UAS_DP_08										X	
SR_UAS_DP_09			X		X						
SR_UAS_DI_01		X									
SR_UAS_DI_02		X									
SR_UAS_DI_03		X						X			
SR_UAS_TC_01		X	X			X					
SR_UAS_TC_02		X	X			X					
SR_UAS_CS_01		X	X	X				X			
SR_UAS_CS_02		X	X	X							
SR_UAS_CS_03			X	X				X			
SR_UAS_CS_04			X	X							
SR_UAS_PS_01						X					
SR_UAS_PS_02						X		X			
SR_UAS_PS_03						X		X			
SR_UAS_PS_04						X		X			

	O.Authentication	O.Data_Integrity	O.Confidentiality	O.Tracing	O.Access	O.Sys_Integrity	O.CI_Standard	O.BE_Standard	O.Physical	O.Replay	O.Availability
SR_UAS_PS_05						X					
SR_UAS_PS_06						X					
SR_UAS_PS_07						X					
SR_UAS_PS_08			X			X					
SR_UAS_PS_09						X					
SR_UAS_SA_01					X						
SR_UAS_SA_02					X				X		
SR_UAS_SA_03					X						
SR_UAS_AV_01											X
SR_UAS_AV_02											X
SR_UAS_AV_03											X
SR_UAS_AV_04											X
SR_UAS_AV_05											X
SR_UAS_SM_3			X	X							
SR_UAS_SV_01		X	X			X					

6.4 Application Services

6.4.1 Requirements

This section defines the security requirements of the FutureID Application Integration Services (AIS). For a better overview, the requirements are grouped according to subtopics.

Identification and Authentication:

SR_AIS_IA_1 The AIS **must** use authentication protocols for communicating with Application Services (AS). The components **must** mutually authenticate using these protocols. The AS **must** use a particular protocol that must ensure that both components are mutually authenticated.

SR_AIS_IA_2 The AIS **must** use authentication protocols such as SAML – Holder of Key [SAML-HoKAP] for communicating with the Federation Service (FS). The components **must** mutually authenticate using these protocols.

SR_AIS_IA_3 The AIS **must** require a re-authentication with AS and FS of a user in case of following events: session inactivity of more than 5 minutes, loss of connectivity to FutureID server, device authentication error, data integrity error, failure in establishing a trusted channel, session termination, and session ended.

SR_AIS_IA_4 The AIS **must** provide security mechanisms to protect the access to sensitive information data from unauthorized access.

SR_AIS_IA_5 When the AIS transmits the user credentials, the transmission **must** be secured. The communication of the credentials between the AIS-AS and the AIS-FS **should** use protocols such as SAML-Holder of Key [SAML-HoKAP].

SR_AIS_IA_6 The AIS **must** maintain the following security attributes of a session belonging to a user after an authentication to the FutureID server has occurred: status of authentication (successful/ unsuccessful/ongoing), trustworthiness level of authentication (see: deliverable D35.1).

SR_AIS_IA_7 The AIS **should** allow no further action until the user has been successfully identified by the FutureID server.

Data Protection:

SR_AIS_DP_01 The AIS **must** ensure that any security relevant data like authentication data, eID data, assurance level, log files are protected. These data **must not** be modified or deleted. The AIS **must** ensure the integrity of data sent to FS.

SR_AIS_DP_02 In case that the AIS, the AS or the FS detect a data integrity error, the AIS **must** close the session and wait for a re-authentication of the user.

SR_AIS_DP_03 Information flow **must not** occur between AIS and FS and between AIS and AS without establishing a trusted path except the information to establish a trusted path (i.e. the information for mutual authentication).

SR_AIS_DP_04 The AIS **must** maintain the confidentiality of data that are being sent through it.

SR_AIS_DP_05 The AIS **must** enforce that all eID data and meta-data which is transferred between AIS and FS and between AIS and AS are encrypted.

SR_AIS_DP_06 The AIS **must** ensure that any security or privacy relevant data from a previous session is deleted and made unavailable as soon as the session is closed or the communication is interrupted.

SR_AIS_DP_07 The AIS **must** enforce that the transmission of eID data and meta-data is protected from replay errors. The AIS **must** be able to detect whether replay has occurred. In this case the AIS **must** terminate the current session and request a re-authentication of the user.

Trusted channel:

SR_AIS_TC_01 The AIS **must** provide a communication channel between itself and Federation Services (FS), and between AIS and AS, that provides assured identification of its end points and protection of the channel data from eavesdropping.

SR_AIS_TC_02 The communication between AIS FS and AS **must** be via trusted channel in order to maintain the confidentiality of the sent data.

SR_AIS_TC_03 The communication between AIS and FS **must** use protocols such as SAML – Holder of Key together with mechanisms that will enhance the resistance against replay attacks [SAML-SSTC].

Cryptographic Support:

SR_AIS_CS_01 The AIS **must** generate cryptographic keys for communicating with FS and AS according to SR_CLI_IA_3.

SR_AIS_CS_02 The AIS **must** ensure that no secret cryptographic keys can be accessed by users or by other software outside the FutureID components.

SR_AIS_CS_03 The AIS **must** delete all temporarily used keys (e.g. session keys) as soon as they are not needed anymore.

SR_AIS_CS_04 For the communication with the FS and the AS, the AIS **should** use protocols (such as SAML HoK [SAML-HoKAP] protocol in case AIS-FS communication), that provide a cryptographically strong alternative using X.509 certificates.

Protection of AIS Security Functionality:

SR_AIS_PS_01 The AIS **should** verify its own integrity during start up and re-start. AIS **should** verify the integrity of its software, checking that no modified software can run within it.

SR_AIS_PS_02 The AIS **must** request acknowledgements from FS and AS for each data transmission on a trusted channel and **must** track the status of each data transmission.

AIS Access:

SR_AIS_CA_01 Any unnecessary access **should** be avoided. The AIS **must** restrict a limit of one session per user and **must not** have more than one session per user at the same time.

SR_AIS_CA_02 The AIS **must** reject connections with unsuccessful authentication.

SR_AIS_CA_03 The AIS **must** be implemented on a site with appropriate physical site security measures in order to prevent unauthorized physical access.

Security Management:

SR_AIS_SM_01 All temporary security attributes (eID data, trust level, authentication status) **must** be deleted after a session has been ended.

Security Audit:

SR_AIS_SA_01 The AIS **must** perform the following actions upon detection of a potential security violation: session termination, residual data deletion in compliance with SR_AIS_DP_06, key destruction in compliance with SR_AIS_CS_03 and security attribute expiration in compliance with SR_AIS_SM_01.

Availability

SR_AIS_AV_01 The AIS **must** ensure the operation of the failure state reporting to other FutureID components when software failures occur in one of its modules.

SR_AIS_AV_02 The AIS **must** assign a priority to each subject in the security functionality. It **must** ensure that access to all sharable resources is mediated on the basis of the assigned priority.

SR_AIS_AV_03 The AIS **must** enforce maximum quotas for memory space, storage space and CPU load that each authentication session can use during the identity federation procedure.

SR_AIS_AV_04 The AIS **should** provide sufficient through-put to offer its services under high load, with significant contingency to spare for unexpected events.

SR_AIS_AV_05 The modules of the AIS **should** provide sufficient resistance against denial-of-service attacks.

6.4.2 Mapping Requirements vs. Objectives

	O.Authentication	O.Data_Integrity	O.Confidentiality	O.Tracing	O.Access	O.Sys_Integrity	O.CI_Standard	O.BE_Standard	O.Physical	O.Replay	O.Availability
SR_AIS_IA_1	X										
SR_AIS_IA_2	X									X	
SR_AIS_IA_3	X	X			X						
SR_AIS_IA_4			X		X						
SR_AIS_IA_5	X		X								
SR_AIS_IA_6	X										
SR_AIS_IA_7	X				X						
SR_AIS_DP_01		X									
SR_AIS_DP_02	X	X									
SR_AIS_DP_03	X		X	X							
SR_AIS_DP_04			X								
SR_AIS_DP_05			X	X							
SR_AIS_DP_06			X	X							
SR_AIS_DP_07										X	

	O.Authentication	O.Data_Integrity	O.Confidentiality	O.Tracing	O.Access	O.Sys_Integrity	O.CI_Standard	O.BE_Standard	O.Physical	O.Replay	O.Availability
SR_AIS_TC_01		X	X	X		X					
SR_AIS_TC_02			X								
SR_AIS_TC_03										X	
SR_AIS_CS_01		X	X	X							
SR_AIS_CS_02		X	X	X							
SR_AIS_CS_03			X	X				X			
SR_AIS_CS_04			X	X				X			
SR_AIS_PS_01						X					
SR_AIS_PS_02						X					
SR_AIS_CA_01					X						
SR_AIS_CA_02					X						
SR_AIS_CA_03					X				X		
SR_AIS_SM_01					X						
SR_AIS_SA_01					X						
SR_AIS_AV_01											X
SR_AIS_AV_02											X
SR_AIS_AV_03											X
SR_AIS_AV_04											X
SR_AIS_AV_05											X

Figure 5: Mapping of AIS security requirements vs. security objectives

6.5 Overall System Requirements

FutureID also needs to cover security requirements for the system-of-systems of the entire infrastructure. For instance, the Broker Service could become a single point of failure due to its exposed position in the structure of the system as a whole. The measures required to protect its security (confidentiality, integrity, availability) are not necessarily properties of the Broker Service itself. They need to be specified globally for the system as a whole.

Since it is unclear at this time within which specific context FutureID will be used in future, organizational policies like security certifications, security audits and compliance to security standards will not be

addressed in this document. Those requirements will depend strongly on the actual application and model of operation (e.g. private/business use, governmental use, types of services, etc.) and have to be defined within a concrete operational context.

6.5.1 Threats

While the treats that were defined in section 5.4 mainly refer to individual system components and the communication between those components, some further threats arise on a system level when considering the FutureID components as a whole. Therefore, for deriving requirements of this chapter, further system threats are defined:

T.S.Integrity An attacker violates the integrity of the overall system by attacking single-point-of-failures or bringing the overall system into an undefined security state. An integrity loss could propagate through the system by this kind of attack.

T.S.Availability An attacker conducts a Denial-of-Service attack on one or more system components which may act as a single-point-of-failure and makes the whole system services unavailable.

T.S.Confidentiality An attacker successfully violates confidentiality of one system component and is thus enabled to attack the confidentiality of other system components.

T.S.Abuse An attacker abuses overall system functionality by attacking the availability of system services or by creating undefined system failure states.

6.5.2 System Objectives

As a result of the threats defined in the last section, several system objectives can be derived for the overall system:

O.S.Integrity The overall FutureID system must ensure that integrity failures within one system component will not lead to integrity failures in other components.

O.S.Confidentiality The overall FutureID system must ensure that confidentiality failures within one system component will not lead to confidentiality failures in other components.

O.S.Availability The overall FutureID system must ensure the highest possible availability of all system components and their services. In case that the availability cannot be fully ensured, the system has to reach a well-defined failure mode with controlled reduced functionality.

O.S.Logging The overall FutureID system has to establish a logging and auditing infrastructure that is able to audit system and component failures, to detect suspicious system behaviour and that can be used to trigger according alerts and countermeasures to maintain secure system functionality.

6.5.3 Mapping threats vs. objectives

The following figure shows how the system threats are addressed by the system objectives:

	O.Integrity	O.Confidentiality	O.Availability	O.Logging
T.S.Integrity	x			x
T.S.Availability			x	
T.S.Confidentiality		x		x
T.S.Abuse	x	x	x	x

6.5.4 Requirements

Based on the system objectives defined in the previous section, the following system requirements can be derived:

Logging/Auditing

SR_SYS_AU_01 For audit events resulting from actions of identified users, all FutureID backend components **must** be able to associate each event with the identity of the user that caused the event, in compliance with the FutureID privacy requirements.

SR_SYS_AU_02 The FutureID backend components **must** be able to apply a set of rules in monitoring the audited events and based upon these rules indicate a potential security violation.

SR_SYS_AU_03 The FutureID backend components **should** be able to maintain profiles of system usage in compliance with the privacy requirements, that allow the detection of any suspicious user activity. In case of detection of a suspicious activity, an alert to the

system administrator **should** be triggered. Depending on the level of severity, a user authentication **may** be blocked until the detected issue is resolved.

SR_SYS_AU_04 The FutureID backend components **should** have available a heuristic method to detect well known attacks and intrusion scenarios. Upon detection, the affected component **should** inform the other components about the security violation and terminate further service activities. Additionally, it **should** trigger an alert to the system administrator.

SR_SYS_AU_05 All FutureID components that generate audit records **must** prohibit all entities read access to the records except for those entities that have been granted explicit access.

SR_SYS_AU_06 Access to all audit records by FutureID components or system administrators **should** be recorded and stored with integrity protection in an access-restricted storage space.

System Availability:

SR_SYS_DE_01 The system of systems that makes the FutureID infrastructure **must not** exhibit a single-point-of failure.

SR_SYS_DE_02 The FutureID Infrastructure **must** implement well-defined failure modes and modes of reduced functionality.

SR_SYS_DE_03 The FutureID infrastructure **must** provide a fail-over mechanism in case of a failure.

SR_SYS_DE_04 In case of a failure, the FutureID infrastructure **must** offer a graceful degradation to an emergency mode that maintains critical functions.

SR_SYS_DE_05 The FutureID infrastructure **must** provide redundancy for key components, such as the Broker Service.

System Confidentiality:

SR_SYS_CO_01 Any confidentiality loss within one system component **must not** lead to confidentiality issues in other system components.

System Integrity:

SR_SYS_IN_01 A loss of integrity within one system component **must not** lead to an integrity loss in another component or to the loss of overall system integrity.

SR_SYS_IN_02 The FutureID infrastructure components **must** establish a mutual synchronisation of system timers to ensure that timeouts and time-restricted token validities are enforced correctly.

SR_SYS_IN_03 The integrity of system time **should** be checked regularly to detect any tampering with time settings.

6.5.5 Mapping requirements vs. objectives

	O.Integrity	O.Confidentiality	O.Availability	O.Logging
SR_SYS_AU_01				x
SR_SYS_AU_02				x
SR_SYS_AU_03	x			x
SR_SYS_AU_04	x			x
SR_SYS_AU_05				x
SR_SYS_AU_06	x			x
SR_SYS_DE_01	x		x	
SR_SYS_DE_02			x	
SR_SYS_DE_03			x	
SR_SYS_DE_04			x	
SR_SYS_DE_05			x	
SR_SYS_CO_01		x		
SR_SYS_IN_01	x			

	O.Integrity	O.Confidentiality	O.Availability	O.Logging
SR_SYS_IN_02	x	x		
SR_SYS_IN_03	x			

7. Conclusions

For FutureID to act as a trusted service it is essential to establish a high level of security for all FutureID services. Therefore, this deliverable has analyzed the security problem definition of the FutureID system components, their distributed interaction and the threats to the overall system performance. Components of the Common Criteria methodology were used to conduct this analysis, yet in a less formal way.

Maintaining security for all components and the overall system is an important prerequisite for establishing trust relations between the user and FutureID services on the one hand and between service providers and FutureID services on the other hand. In addition, security is also one of the prerequisites for maintaining user privacy.

Along the classical protection goals of confidentiality, integrity, availability and authenticity, specific security requirements have been derived for each system component. Associating the requirements to system components on the one hand generates many duplications of requirements for other components but on the other hand also allows an implementer to concentrate only on one chapter for one component.

Due to the distributed nature of the FutureID infrastructure, further threats may arise for the system as a whole when certain attack scenarios are considered. These scenarios may lead to security violations within the overall system, potentially without affecting the security of an individual component. Therefore, threats to the overall system have been identified and corresponding requirements have been derived.

This deliverable should therefore serve as a guideline for implementers and should form the basis of more specific requirements for the various system components.

8. References

[BSI102] Technical Guideline TR-02102, Cryptographic Methods: Recommendations and key lengths, BSI.

[SAML-HoKAP] Tom Scavo. SAML V2.0 Holder-of-Key Assertion Profile. OASIS Committee Specification 02, 23.01.2010. http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml2-holder-of-key-cs-02.pdf, 2010.

[SAML-SSTC] Jeff Hodges and Chris McLaren, Security and Privacy Considerations for the OASIS Security Assertion Markup Language (SAML), 15.01.2002. https://www.oasis-open.org/committees/security/docs/draft-sstc-sec-consider-03.doc, 2002.

[RFC2119] S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels," IETF RFC 2119. http://www.ietf.org/rfc/rfc2119.txt

[TLS1.1] T. Dierks, E. Rescorla, The Transport Layer Security (TLS) Protocol, Version 1.1, IETF RFC 4346. http://www.ietf.org/rfc/rfc4346.txt