TALLINN UNIVERSITY OF TECHNOLOGY

Faculty of Information Technology

Department of Informatics

Chair of Software Engineering

Analysis of Configuration Management in Federated X-Road Systems

Bachelor's thesis

Student:	Riin Saarmäe
Student code:	IABB111907
Supervisors:	PhD. Ermo Täks
	PhD. Margus Freudenthal

Tallinn 2015

Author's Declaration of Originality

Herewith I declare that this thesis is based on my own work. All ideas, major views and data from different sources by other authors are used only with a reference to the source. The thesis has not been submitted for any degree or examination in any other university.

Author: Riin Saarmäe

_____

(signature)

15.01.2015

Abstract

The aim of this thesis is to implement the analysis of configuration management in federated X-Road systems.

The analysis will find procedural and technical solutions to federating two separate X-Road systems. The solution will allow the organizations belonging to the federated systems to carry out cross-border data exchange, but will not add any additional requirements to the organizations. A conceptual model for federation will be created and the processes involved will be described. Communications between the X-Road components and functionality that must be added to the components to achieve federation-capabilty will be specified.

The analysis will be implemented using the Rational Unified Process methodology.

As a result of the analysis, a conceptual model describing the entities participating in creating and maintaining a federation relationship and the relationships between the entities will be created. A business use case model describing the most significant business processes taking place in a federation will be described. The data objects used by the system components will be modeled using class diagrams. The functionality of the system components will be described in use case models. The design and functionality of user interfaces will be specified and user roles and privileges will be described.

The thesis is written in English and contains 70 pages of text, 10 chapters, 11 figures, 8 tables, 1 annex.

Annotatsioon. Födereerunud X-tee süsteemide konfiguratsioonihalduse analüüs

Töö eesmärk on födereerunud X-tee süsteemide konfiguratsioonihalduse analüüs, mille raames leitakse protseduurilised ja tehnilised lahendused föderatsiooni loomiseks kahe X-tee süsteemi vahel. Leitav lahendus peab võimaldama piiriülest andmevahetust födereerunud süsteemidesse kuuluvate organisatsioonide vahel, kuid ei tohi andmevahetuses osalevatele organisatsioonidele kaasa tuua lisanõudmisi.

Töö käigus luuakse X-tee süsteemide föderatsiooni kontseptuaalne mudel, kirjeldatakse seonduvad protsessid ning spetsifitseeritakse X-tee komponentidele föderatsioonisuutlikkusega lisanduv funktsionaalsus ja komponentidevahelised andmevahetusprotsessid.

Analüüs viiakse läbi kasutades Rational Unified Process metoodikat.

Analüüsi tulemusena luuakse kontseptuaalne olemimudel, mis kirjeldab föderatsioonisuhte realiseerimisel osalevaid olemeid ja olemitevahelisi seoseid. Olulisimad föderatsioonis aset leidvad talitlusprotsessid kirjeldatakse talitlusmallimudelis. X-tee süsteemi komponentide poolt konfiguratsioonihalduses kasutatavad andmeobjektid modelleeritakse klassidiagrammidena komponentide kontseptuaalsetes andmemudelites. Süsteemi komponentide funktsionaalsus esitatakse kasutusmallimudelites. Kirjeldatakse süsteemi komponentide kasutajaliideste disain ja funktsionaalsus ning kasutajarollid ja rollidele omistatud privileegid.

Lõputöö on kirjutatud inglise keeles keeles ning sisaldab teksti 70 leheküljel, 10 peatükki, 11 joonist, 8 tabelit ja 1 lisa.

List of Figures

2.1 Components of an X-Road infrastructure. 13

3.1 Communication in federated X-Road systems. 17

4.1 Conceptual diagram for configuration management. 20

5.1 Business use case diagram for establishing federation relationship. 25

6.1 Conceptual data diagram for central server. 28

7.1 Conceptual data diagram for security server. 33

8.1 Use case diagram for central server. 38

9.1 Use case diagram for security server. 47

10.1 User interface design draft for central server configuration sources sub-view. . . 56

10.2 User interface design draft for central server trusted anchors sub-view. 59

10.3 User interface design draft for security server configuration anchor section. . . 61

List of Tables

6.1 Central server data model entities for configuration distribution. 28
6.2 Central server data model entity attributes for configuration distribution. 29

7.1 Security server data model entities for configuration distribution. 34
7.2 Security server data model entity attributes for configuration distribution. . . . 34

10.1 Central server user roles and user groups. 54
10.2 Central server user roles and user action privileges. 54
10.3 Security server user roles and user groups. 60
10.4 Security server user roles and user action privileges 61

Contents

1 Introduction **10**

1.1 Methods 11

1.2 Overview of the Thesis 11

2 Overview of the X-Road System **12**

2.1 Architecture 12

2.2 Components 13

3 Federation of X-Road Systems **16**

3.1 Purpose 16

3.2 Requirements 16

3.3 Technical Solution 17

3.4 Analysis 18

4 Conceptual Model **20**

5 Business Use Case Model **23**

5.1 Establishing Federation Relationship 23

5.2 Ending Federation Relationship 26

5.3 Updating Configuration Anchor 26

6 Conceptual Data Model for Central Server **28**

7 Conceptual Data Model for Security Server **33**

8 Central Server Use Case Model **37**

8.1 Actors 37

8.2 General System Error Handling 37

8.3 User-System Use Cases 38

8.3.1 UC CS01: View Configuration Source . . . 38
8.3.2 UC CS02: Download Configuration Source Anchor . . . 39
8.3.3 UC CS03: Re-Create Configuration Source Anchor . . . 39
8.3.4 UC CS04: Upload Optional Configuration Part . . . 39
8.3.5 UC CS05: Download Configuration Part . . . 40
8.3.6 UC CS06: Update Central Server Address . . . 40
8.3.7 UC CS07: Log In to Security Token . . . 40
8.3.8 UC CS08: Log Out of Security Token . . . 41
8.3.9 UC CS09: Add Configuration Source Signing Key . . . 41
8.3.10 UC CS10: Activate Configuration Source Signing Key . . . 42
8.3.11 UC CS11: Delete Configuration Source Signing Key . . . 42
8.3.12 UC CS12: View Trusted Anchors . . . 43
8.3.13 UC CS13: Upload Trusted Anchor . . . 43
8.3.14 UC CS14: Download Trusted Anchor . . . 44
8.3.15 UC CS15: Delete Trusted Anchor . . . 44
8.3.16 UC CS16: View Global Error Messages . . . 45
8.4 System Use Cases . . . 45
8.4.1 UC CS17: Generate Configuration Anchor . . . 45
8.4.2 UC CS18: Generate Configuration . . . 45
8.4.3 UC CS19: Download Configuration . . . 46

9 Security Server Use Case Model 47
9.1 Actors . . . 47
9.2 General System Error Handling . . . 48
9.3 User-System Use Cases . . . 48
9.3.1 UC SS01: View Configuration Anchor . . . 48
9.3.2 UC SS02: Download Configuration Anchor . . . 48
9.3.3 UC SS03: Upload Configuration Anchor . . . 49
9.3.4 UC SS04: View Global Error Messages . . . 49
9.4 System Use Cases . . . 49
9.4.1 UC SS05: Update Configuration . . . 49
9.4.2 UC SS06: Download Configuration . . . 50
9.4.3 UC SS07: Access Global Configuration . . . 51
9.4.4 UC SS08: Update System State . . . 52

10 User Interfaces **53**

10.1 User Roles and Privileges 53

10.1.1 Central Server User Roles, Groups and Privileges 53

10.2 Central Server User Interface Specification 55

10.2.1 Configuration Management View 55

10.2.2 Internal/External Configuration Sub-View 56

10.2.3 Trusted Anchors Sub-View 59

10.3 Security Server User Interface Specification 60

10.3.1 Security Server User Roles, Groups and Privileges 60

10.3.2 Configuration Anchor Section in the System Parameters View 61

10.4 Global Error Messages 62

Bibliography **65**

A Screenshots of Graphical User Interfaces **68**

A.1 Central Server Configuration Management: Internal Configuration 68

A.2 Central Server Configuration Management: Trusted Anchors 69

A.3 Security Server System Parameters: Configuration Anchor 69

A.4 Central Server: Global Error Messages 70

Chapter 1

Introduction

The Estonian national development plan for information and communication technology [1] states, that by 2020, Estonia should be exchanging cross-border e-services with seven foreign countries. As a step towards this goal, the project "Development of X-Road system supporting cross-border services" was initiated by the Estonian Information System Authority [2]. The goal of the project is to develop the capability of facilitating cross-border e-services into the X-Road system. One of the main stakeholders in this project is the Finnish Government. Finland is presently in progress of implementing a data exchange layer in Finland that is based on the Estonian X-Road [3]. First pilots for cross-border services between Estonian and Finnish tax agencies have been conducted.

In July 2014, the Estonian Information System Authority declared a public procurement for development of X-Road system supporting cross-border services. The procurement contract was awarded to and carried out by Cybernetica AS.

The author of this thesis works as system analyst in Cybernetica AS and was responsible for conducting the system analysis for this project.

The analysis of the system was implemented in two phases (as requested in the public procurement [2]) First, an initial analysis was carried out to find technical and procedural solutions for supporting cross-border services in the X-Road system. The artifacts created during the initial analysis were the business use case model, conceptual model and first drafts of the component use case models and user interface designs; capturing the main processes, data objects and functionality proposed for the system. The results of the initial analysis were presented to the Estonian Information System Authority to be evaluated and used for specification of the scope of the system under development. Subsequently, the analysis was further detailed with system component data models, also the component use case models and user interface specifications were enhanced to serve as input to the realization of the system.

1.1 Methods

Cybernetica AS uses the iterative Rational Unified Process (RUP) [4] as the software development process framework [5]. The RUP project life-cycle consists of four phases: inception, elaboration, construction and transition. The main objectives of the inception phase are definition of the scope of the project and the main business use cases, requirements analysis and risk assessment. In the elaboration phase, the system's architectural foundation is defined, the functional requirements are captured in the use case model, the data objects are detailed in the conceptual data models and realization of the main use cases is started. The construction of the system is completed in the construction phase and delivered to the users in the transition phase. The artifacts created in the analysis process are further detailed in section 3.4.

The Unified Modeling Language (UML) is used to visualize the design of the system [6, 7]. In Cybernetica AS, Visual Paradigm [8] software is used for system modeling.

The use cases are mainly written at the user goal level [9], describing the "one sitting" user-system interaction where the main success scenario ends with the user fulfilling a business goal. As the main functionality of the system (configuration distribution) requires no direct user input once the system is configured by the user, some of the most important system-system and internal system use cases are also described.

The graphical user interface design drafts were drawn using Pencil Project [10] software.

1.2 Overview of the Thesis

Chapter 2 gives a brief overview of the X-Road system (as-is description of the system). Chapter 3 outlines the purpose, requirements and technical solutions for modifications of the system (to-be description of the system). The analysis of the system under development is detailed in chapters 4-10. The analysis consists of conceptual model and business use case model of the system; and conceptual data models, use case models and user interface specifications for the system components.

Chapter 2

Overview of the X-Road System

The Estonian X-Road was launched in 2001 and has been growing and evolving rapidly ever since – both in number of users and in technological advancements. By 2013 the X-Road has interconnected more than 900 organizations, public registers and databases that are providing and using more than 2000 services. The X-Road handles more than 300 million transactions per year [11]. In Estonia, pursuant to the Public Information Act [12], the exchange of data with the databases belonging to the state information system and between the databases belonging to the state information system must be carried out through the X-Road system [13]. The X-Road system currently in production in Estonia is X-Road version 5. Since 2012 [14], Estonian Information System Authority in collaboration with Cybernetica AS, Estonian eHealth Foundation, the Finnish Government and other partners, has been developing the next generation of the system – X-Road version 6. Since version 6 is not backwards-compatible with version 5, a transitional version of X-Road system – version 5.5 – is developed parallel to version 6 to facilitate the transition from version 5 to version 6 in Estonia. Version 5.5 encompasses both the version 5 and version 6 systems allowing cross-usage of the systems. The public pilot for version 5.5 (in Estonia) and version 6 (in Finland) is planned to commence in 2015 (Heiko Vainsalu, personal communication, 2014). As X-Road version 5 will be replaced by version 6, new functionality (including the capability for cross-border data exchange) will only be added to X-Road version 6.

The following sections describe the main architectural principles and system components of the X-Road version 6.

2.1 Architecture

The X-Road system is a distributed, secure, unified web-services based inter-organizational data exchange framework [11].

The design of the X-Road is based on the following principles [15].

- Distributed architecture – X-Road is a completely distributed, resilient system with distributed management. X-Road does not centralize the data exchange and does not change the ownership of the data.
- Heterogeneous integration – X-Road connects information systems built on any IT platform. X-Road does not prescribe any tools and technologies for intra-organizational use.
- End-to-end confidentiality – the participants of the X-Road communicate directly with each other. The messages are encrypted at the end points and thus the exchanged data is not visible to any external parties.
- Authentication and non-repudiation – all the messages (both the requests and the responses) are signed by the originating entity using a qualified certificate. The signing process uses secure signature creation devices (SSCD). The signed messages are time-stamped and logged so that the validity of the signatures can be verified at a later date.
- Reliability – the system does not have a single point of failure. All components of the system can be made redundant for high resiliency against failures and attacks. Components that are available over shared or public network employ protective measures against denial of service (DoS) attacks.
- Based on open standards – X-Road is based on open standards such as HTTP, SOAP [16], WSDL [17], MIME [18], X.509 [19], TLS [20], RFC3161 [21], and RFC6960 [22].

2.2 Components

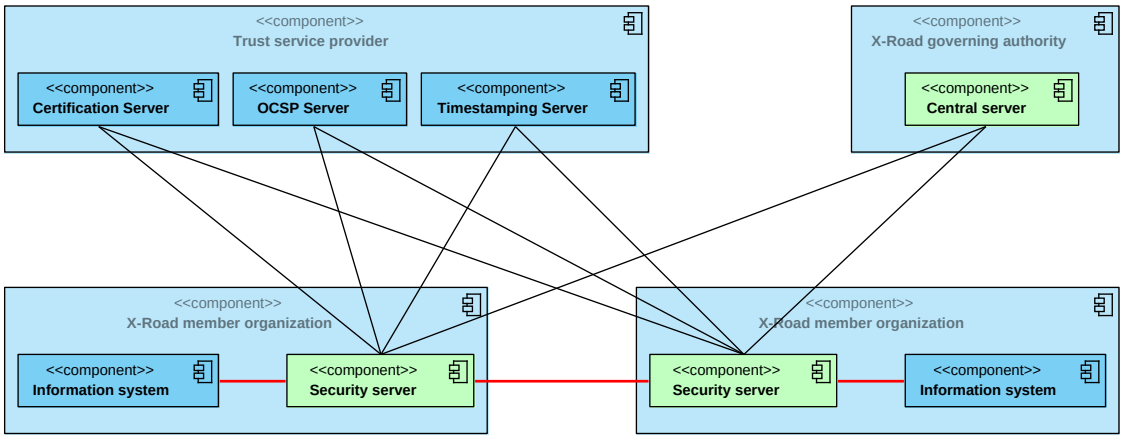


Figure 2.1: Components of an X-Road infrastructure.

Figure 2.1 displays the logical structure of the X-Road system: organizations, servers and network communication. The X-Road core components are shown in green. The message flow between information systems participating in data exchange is shown in red. The X-Road organizational infrastructure is composed of the following principal parties: the X-Road governing authority, one or more accredited trust service providers, X-Road member organizations. [23, 15]

The X-Road governing authority is an organization responsible for creating and maintaining an X-Road infrastructure instance. The governing authority

- appoints a number of trust service providers to provide trust services to the X-Road member organizations,
- approves and registers X-Road member organizations and the associated security servers to the X-Road infrastructure.

The central server maintains a database of X-Road member organizations and associated access points (security servers), approved trust services, and other infrastructural configuration elements. The central server is responsible for distributing the infrastructural configuration to security servers where it is used for setting up secure data exchange channels between member organizations and also as a base for service lookup and discovery.

X-Road member organization is a legal entity that provides or consumes web-services over the X-Road. An X-Road member organization

- manages a security gateway,
- obtains a certificate for the security server from an approved trust service provider. The certificate is used for establishing secure connections between security servers participating in data exchange;
- obtains a qualified certificate for digital signing of the exchanged messages from an approved trust service provider,
- contracts with an approved trust service provider for a time-stamping service,
- uses an information system providing or consuming web-services via the security server.

The security server mediates and controls the secure communication between information systems. Security server also provide metaservices for service lookup and discovery.

Trust service provider provides standard trust services:

- certificates for digital signatures and web servers,

- certificate validity confirmation service using OCSP protocol,
- time-stamping service using RFC 3161 protocol.

The protocol used for communication between security servers is designed based on the following requirements [15]:

- the protocol must be suitable for protecting web-service requests and responses;
- the protocol must allow exchanging payloads with unlimited size; the protocol must allow signing all payloads (including the payloads with unlimited size);
- the protocol must support signing multiple messages with the same signing operation in order to support low-speed secure signature creation devices (SSCD);
- the protocol must use on-line public key infrastructure (PKI) services in a scalable way.

The X-Road security gateway protocol is based on mutually authenticated HTTPS. Message content is encapsulated in MIME multiparts. Messages are signed using XAdES [24] and ASiC [25] data formats. Batch signatures [26] are used to achieve good performance even when using low-speed SSCDs. Security servers aggregate time-stamping requests and cache OCSP responses to reduce load on on-line PKI services. [15, 23]

Chapter 3

Federation of X-Road Systems

3.1 Purpose

The objective of the project "Development of X-Road system supporting cross-border services" is to develop a solution for a network of X-Road systems situated in different countries exchanging cross-border services. The data exchange projects in many countries encompass the need to exchange data not only internally but also internationally [1], thus, a solution allowing for cross-border message exchange between X-Road systems must be developed. The project will result in creating trust federation between X-Road systems, where members belonging to different X-Road infrastructures will be able to use services provided by members of other (federated) systems, making the X-Road systems interoperable. [2]

In 2013, the Estonian and Finnish governments agreed upon sharing and joint development of X-Road solution, leading to the implementation of a data exchange layer in Finland that is akin to the Estonian X-Road [27, 3]. The initial pilot projects for cross-border e-services will be carried out between the Estonian and Finnish tax boards [27].

3.2 Requirements

The main requirements for developing federation-capability for the X-Road system stated in the public procurement [2] are the following.

- The solution for cross-border services can not require changes in the structure or functionality of X-Road services.
- Use of cross-border services can not add additional requirements to the X-Road members.
- Federation must be bilateral - the federation agreement has two counterparts.

- X-Road system retains its autonomy when entering in a federation relationship. Governing agencies of the federated systems remain in control of the security policy of their native system.
- Establishing the federation relationship is carried out at the governing agency's level, the members do not have to make any additional arrangements.

3.3 Technical Solution

To accommodate the requirements, the following technical solution was proposed [28].

Figure 3.1 describes the organization-level information flow in federated X-Road systems. The light blue lines indicate trust relationships, darker blue lines indicate the use of trust services, green lines stand for configuration distribution and red lines for X-Road messages.

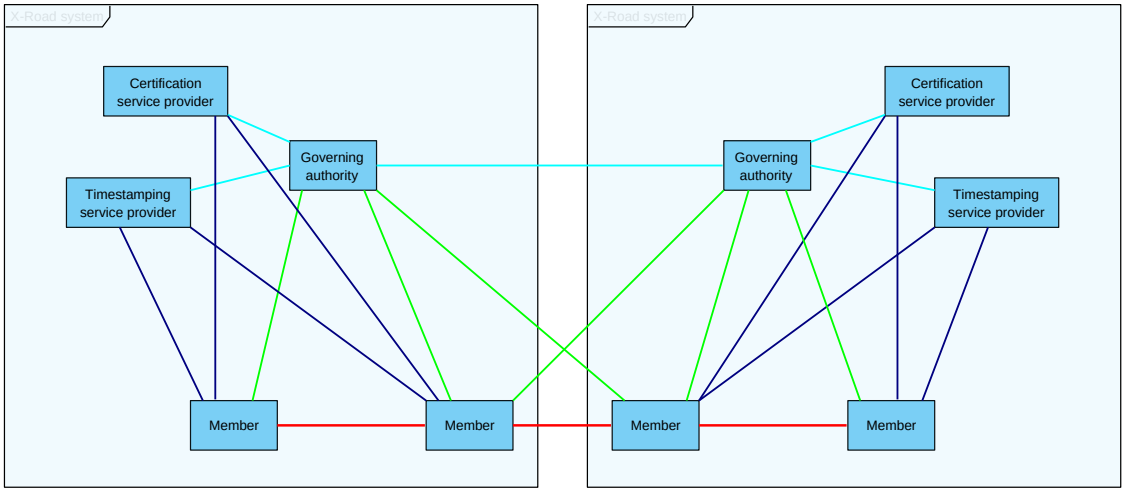


Figure 3.1: Communication in federated X-Road systems.

The development of federation-capable X-Road system includes the following functional changes to the existing system.

1. The configuration distributed by the X-Road governing authority is divided into internal and international (external) parts. The international configuration consists of data needed for cross-border message exchange:
    - approved trust services;
    - central services;
    - global groups;

- X-Road members, member's subsystems, member's security servers;
- authentication certificates and clients registered to security servers;
- security policy parameters for message exchange.

The internal configuration consists of data relevant only to the native X-Road system:

- parameters for central server's management services;
- information about the configuration sources of X-Road systems federated with the native system;
- internal security policy parameters.

2. The protocol used for configuration distribution is supplemented with the possibility to download only the international part of an X-Road system's configuration, to allow the same protocol to be used for both internal and international configuration. For establishing a federation relationship, the parameters for downloading the international configuration of the federation partner are added to the internal configuration of both federating systems.

3. A new system component – configuration proxy – will be developed. Configuration proxies may be used to increase performance and reliability in situations where the network connection to the primary configuration source is slow or unreliable, also to reduce load to the primary source. For now, the component's functionality is limited with caching, but additional functions may be added later. For instance, a configuration proxy mediating configuration from one X-Road instance to another could be used to modify the configuration by filtering out elements that do not conform to the security policy of the receiving X-Road instance.

4. The existing system components – security server and central server – is complemented with additional functionality for managing configuration.

3.4 Analysis

Next chapters of the thesis present system analysis for the project "Development of X-Road system supporting cross-border services". The analysis was carried out by the author of this thesis.

The analysis consists of following parts.

- The conceptual model describes the entities participating in the configuration management and the relationships between the entities.

- The business use case model defines the key business use cases of the system under development.
- The system component data models describe the data objects used and managed by the components for configuration management and distribution.
- The system component use case models give a detailed description of the user-system interactions. Also, since the primary functionality of the system (once configured by the user) requires no user input, some of the key system use-cases are also described.
- The component user interface specifications define the user interface design and functionality among with user roles and privileges.

To accommodate to the prescribed scope limit of bachelor's thesis, the data models, use case models and user interface specifications are described for two system components – central server and security server, leaving out the configuration proxy. The configuration proxy acts both as a configuration provider and a configuration client. The data model and use case model for the configuration proxy are in essence derivates of central server (acting as a configuration provider) and security server (acting as a configuration client) models. The configuration proxy does not have a graphical user interface, user interactions are carried out using command line interface.

Chapter 4

Conceptual Model

This chapter describes the entities participating in configuration management and the relationships between the entities. Figure 4.1 depicts the entities and relationships, which are further detailed below.

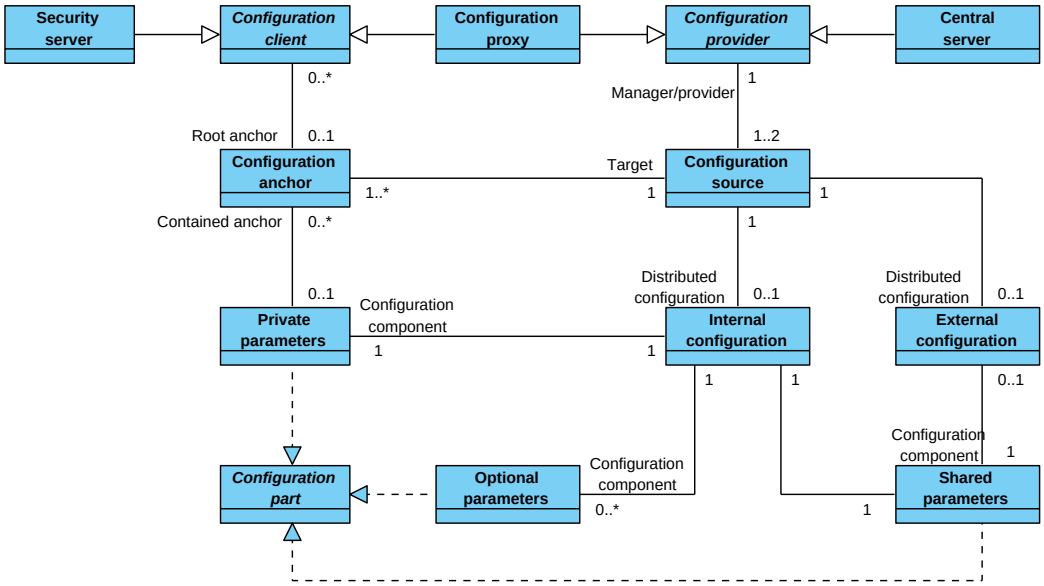


Figure 4.1: Conceptual diagram for configuration management.

Configuration provider is an entity responsible for maintaining and distributing configuration. Configuration provider manages one or two **configuration sources** through which configuration is made available for configuration clients. In an X-Roads system, **central server** and **configuration proxy** act as configuration providers.

Central server is the primary **configuration source** in an X-Road system. Central server always manages an **internal configuration** source (i.e. configuration source distributing the internal configuration) and in addition, an **external configuration** source (i.e. con-

figuration source distributing the external configuration) in case the X-Road system is federation-capable.

Configuration proxy may optionally be used to mediate configuration originating from the **central server** to the **configuration clients**. Configuration proxy manages a single configuration source, that is used to distribute configuration downloaded from another **configuration source**. The configuration mediated by the proxy may either be internal or external configuration, depending on the proxy's purpose.

Configuration source is a component (HTTP server) managed by a **configuration provider**. The configuration distributed by the source can either be **internal configuration** or **external configuration**. The information needed to access and download configuration from a source is contained in the **configuration anchor**.

Internal configuration is distributed by a **configuration source** and is composed of the following configuration parts: **private parameters**; **shared parameters**, and; optionally, other configuration parts that are specific to an X-Road instance - **optional parameters**.

External configuration is distributed by a **configuration source** and only contains the **shared parameters** configuration part.

Private parameters is a **configuration part** that holds system parameters that are only used by **security servers** that are part of the native X-Road system (i.e. the same X-Road system as the **central server** the configuration part originates from). In case of federated X-Road systems, the private parameters contain **configuration anchors** pointing to configuration sources distributing external configuration of federation partners.

Shared parameters is a **configuration part** that holds system parameters that are used both by the security servers of the native X-Road system and by the security servers belonging to X-Road systems federated with the native system.

Optional parameters is an optional configuration part that carries system parameters that have a contextual meaning only to a specific X-Road system installation. For example, the transitional version of the Estonian X-Road system - version 5.5 - requires the use of an optional configuration part (identifiermapping), that contains the translation of X-Road version 5 identifiers to the respective X-Road version 6 identifiers.

Configuration part is an XML file containing system parameters.

Configuration anchor is a set of information that can be used by **configuration clients** to access a **configuration source** and to verify the downloaded configuration. The configuration anchor is distributed as either a separate XML file in case the anchor points to a native configuration source or as a part of **private parameters** in case the anchor points to the configuration source managed by a federation partner.

Configuration client is an entity that uses configuration anchor(s) for downloading configuration from configuration source(s). In an X-Roads system, **security server** and **configuration proxy** act as configuration clients.

Security server is a **configuration client** that uses a root **configuration anchor** to download internal configuration from its native configuration source. Subsequently, in case of federated X-Road systems, the security server uses configuration anchors contained in the private parameters part of the downloaded internal configuration to download the external configuration of X-Road systems federated with the native system.

Chapter 5

Business Use Case Model

5.1 Establishing Federation Relationship

CONTEXT: This use case describes the process of federating two X-Road systems. The process is described from the viewpoint of one X-Road system (native system), the process in the other system (partner system) mirrors the process in the native system. For the federation relation to be functional, both federating systems must carry out the scenario described below.

PRECONDITIONS:

- The governing authorities of the federating X-Road systems have agreed upon the federation conditions and entered into federation agreement.
- Both systems are technically federation-capable.

POSTCONDITION: Readiness for cross-border service exchange between federated X-Road systems.

MAIN SUCCESS SCENARIO:

1. The native system creates a digital configuration anchor file carrying information of its configuration source distributing external configuration.
2. The native system forwards (via out of band means) the created configuration anchor, along with means to validate the integrity of the anchor, to the partner system.
3. The native system receives and validates the configuration anchor of the partner system.
4. The native system integrates the partner system's configuration anchor to internal configuration distributed to the security servers of the native system.
5. The native system distributes (internal and external) configuration.

VARIATIONS:

2,3a. Integrity of the anchor file can be protected against modification by

2,3a1. digitally signing the file or

2,3a2. providing the file digest separately from the anchor file.

ALTERNATIONS:

2a. Using configuration proxy for outgoing configuration:

2a1. The native system sets up a configuration proxy using the created anchor file.

2a2. The native system creates a digital configuration anchor file carrying information of the configuration proxy.

2a3. The native system forwards (via out of band means) the configuration proxy anchor, along with means to validate the integrity of the anchor, to the partner system. Use case continues from step 3.

4a. Using configuration proxy for incoming configuration:

4a1. The native system sets up a configuration proxy using the partners anchor file.

4a2. The native system creates a digital configuration anchor file carrying information of the configuration proxy.

4a3. The native system integrates the configuration proxy's configuration anchor to configuration distributed to the security servers of the native system. Use case continues from step 5.

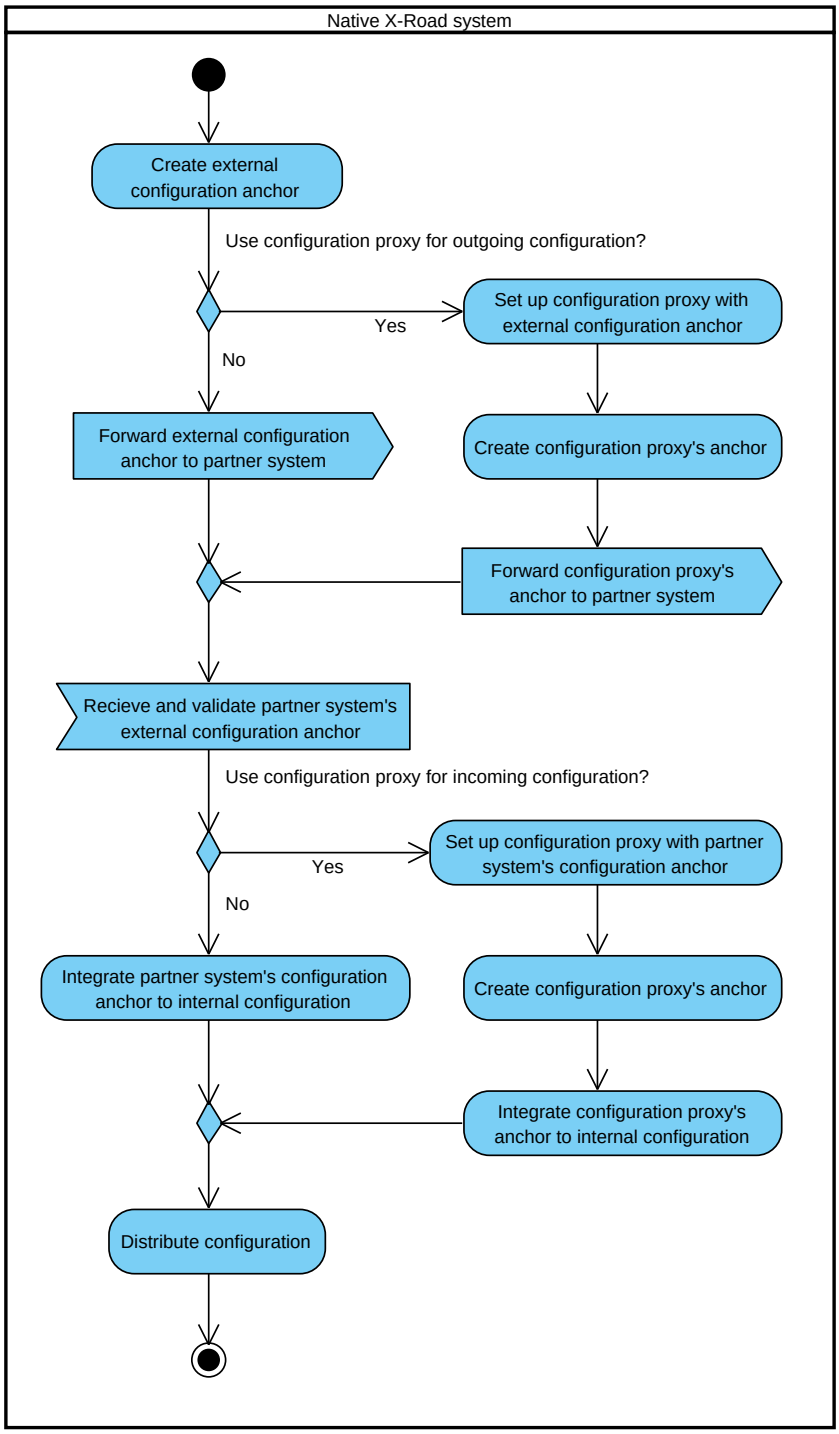


Figure 5.1: Business use case diagram for establishing federation relationship.

5.2 Ending Federation Relationship

CONTEXT: This use case describes the process of ending an established federation relationship between two X-Road systems. The process is described from the viewpoint of one X-Road system (native system), the process in the other system (former partner system) mirrors the process in the native system. The federation relation ceases to be functional (in the means of cross-border service exchange), as soon as one of the systems has carried out the scenario described below.

PRECONDITION: The governing authorities of federated X-Road systems have agreed upon ending the federation relationship.

POSTCONDITION: Cross-border data exchange between former federation partners is rendered impossible.

MAIN SUCCESS SCENARIO:

1. The native system deletes the former partner system's configuration anchor from the internal configuration distributed to the security servers of the native system.
2. The native system distributes configuration.

ALTERNATIONS:

1a. System uses configuration proxy for outgoing configuration.

1a1. The native system dismantles the configuration proxy mediating the outgoing configuration. Use case continues from step 1.

1a. System uses configuration proxy for incoming configuration.

1a1. The native system deletes the configuration proxy's configuration anchor from the configuration distributed to the security servers of the native system.

1a2. The native system dismantles the configuration proxy mediating the incoming configuration. Use case continues from step 2.

5.3 Updating Configuration Anchor

CONTEXT: This use case describes the process of updating configuration anchor in case the attributes of a configuration source the anchor is pointing to have changed. Since the anchor holds means to both accessing the configuration source and verification of downloaded configuration, it is essential to be updated as soon as possible. The use case describes the actions

of a configuration provider responsible for distributing configuration (represented by either an X-Road central server or configuration proxy manager) and of a configuration client who uses the distributed configuration (represented by either an X-Road security server or configuration proxy manager).

PRECONDITION: The attributes (download URL or signing key) of a configuration source are changed.

POSTCONDITION: Configuration clients use the updated anchor for downloading and verifying configuration.

MAIN SUCCESS SCENARIO:

1. Configuration provider creates a digital configuration anchor file corresponding to the changed configuration source.
2. Configuration provider forwards (via out of band means) the configuration anchor, along with means to validate the integrity of the anchor, to configuration client.
3. Configuration client receives and validates the configuration anchor and imports the configuration anchor to the system.

Chapter 6

Conceptual Data Model for Central Server

This chapter describes the entities and their attributes in the X-Road central server that are directly participating in configuration distribution. Figure 6.1 depicts the entities and their attributes, which are further detailed in tables 6.1 and 6.2.

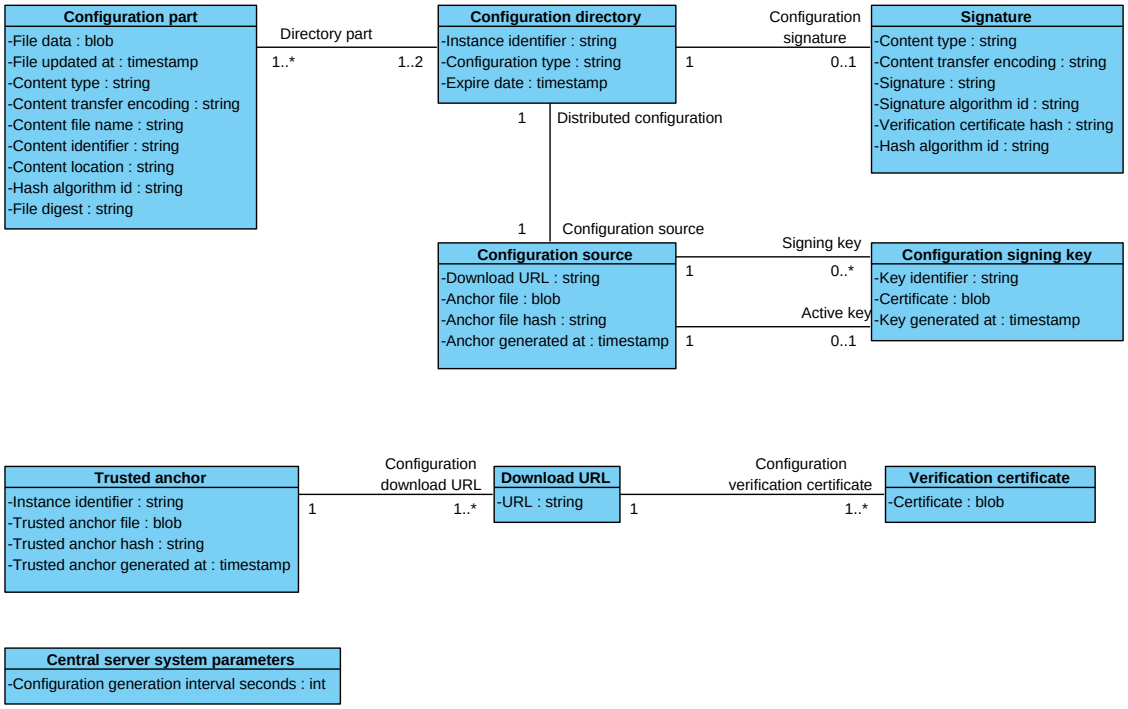


Figure 6.1: Conceptual data diagram for central server.

Table 6.1: Central server data model entities for configuration distribution.

Entity	Description
Configuration source	Component distributing signed configuration.

Configuration signing key	Key that can be used to sign configuration.
Configuration directory	An assembly of configuration parts.
Signature	Digital signature used to protect the integrity of the distributed configuration.
Configuration part	Set of system parameters.
Trusted anchor	Configuration anchor pointing to a configuration source distributing external configuration of a federation partner.
Download URL	URL that can be used by configuration clients to download configuration.
Verification certificate	Certificate used to verify downloaded configuration.
Central server system parameters	Central server inner parameters, not part of distributed configuration.

Table 6.2: Central server data model entity attributes for configuration distribution.

Entity	Attribute	Description
Configuration part	File data	Configuration part file.
Configuration part	File updated at	The date and UTC time in ISO 8601 format [29] of the last update of the configuration part file.
Configuration part	Content file name**	Name of the configuration part file.
Configuration part	Content type*	Content type description. Value must be *"application/octet-stream"*.
Configuration part	Content transfer encoding*	Content encoding description. Value must be *"base64"*.
Configuration part	Content identifier**	Identifies the type of the configuration part. Predefined values are *"PRIVATE-PARAMETERS"* and *"SHARED-PARAMETERS"*. In addition, each X-Road installation using optional configuration parts must define additional content-identifiers for the optional parts.
Configuration part	Content location*	URL that can be used to download the configuration part file.

Configuration part	Hash algorithm id*	Hash algorithm identifier used to calculate the verification certificate hash.
Configuration part	File digest****	Digest of the configuration part. The digest algorithm is specified by the Hash-algorithm-id attribute value. The input to the digest calculation is body of the file that can be downloaded from the URL specified by the Content-location attribute value.
Configuration directory	Configuration part	Configuration part belonging to the configuration directory.
Configuration directory	Signature	Signature given to the configuration directory.
Configuration directory	Configuration type	Type of the configuration. Values are predefined, the attribute value can either be *"internal"* or *"external*".
Configuration directory	Expire date*	End of validity time of configuration. Contains date and UTC time in ISO 8601 format [29].
Configuration directory	Instance identifier***	Identifier of the X-Road instance the configuration part originates from.
Signature	Content type*	Content type description. Value must be *"application/octet-stream"*.
Signature	Content transfer encoding*	Content encoding description. Value must be *"base64"*.
Signature	Signature****	Value of the signature calculated by signature algorithm identified by the Signature-algorithm-id attribute value. Active signing key is used to create the signature.
Signature	Signature algorithm id*	The signature algorithm used to create the signature. Algorithm identifiers listed in XML Signature specification [30], Section 6.4 are supported.
Signature	Verification certificate hash*	The hash of the certificate that was used to sign configuration.

Signature	Hash algorithm id***	Hash algorithm identifier used to calculate the verification certificate hash.
Configuration source	Configuration directory	Configuration directory distributed by the configuration source.
Configuration source	Configuration signing key	Key that can be used to sign the configuration distributed by the configuration source.
Configuration source	Configuration signing key	Active key that is currently used to sign the configuration distributed by the configuration source.
Configuration source	Download URL	HTTP URL used to download configuration provided by the configuration source.
Configuration source	Anchor file	Anchor file respective to the configuration source.
Configuration source	Anchor file hash	Hash of the anchor file, computed using hash function SHA-224 [31].
Configuration source	Anchor generated at	Generation date and time in ISO 8601 format [29] of the anchor file.
Configuration signing key	Key identifier	Identifier of the signing key.
Configuration signing key	Certificate	Self-signed X.509 [19] certificate used as a container for the public key part of the signing key.
Configuration signing key	Key generated at	Generation date and time in ISO 8601 format [29] of the signing key.
Trusted anchor	Download URL	URL described in the anchor.
Trusted anchor	Instance identifier	Identifier of the X-Road instance the anchor originates from.

Trusted anchor	Trusted anchor file	Anchor file containing the following information: • generation time of the anchor file; • instance identifier of this X-Road system; • configuration source's download URL; • certificates of the configuration source's signing keys;
Trusted anchor	Trusted anchor hash	Hash of the anchor file.
Trusted anchor	Trusted anchor generated at	Generation date and time of the anchor file.
Download URL	Verification Certificate	Certificate that can be used to verify the configuration downloaded from the download URL.
Download URL	URL	HTTP URL pointing to a configuration source.
Verification Certificate	Certificate	Public key that can be used to verify the configuration downloaded from the configuration source pointed by the trusted anchor.
Central server system parameters	Configuration generation interval seconds	System parameter defining the interval of configuration generation.

* Used as an obligatory MIME header specified in the Protocol for Downloading Configuration [32].

** Used as an optional MIME header specified in the Protocol for Downloading Configuration [32].

*** Used in MIME header value specified in the Protocol for Downloading Configuration [32].

**** Used in MIME message body specified in the Protocol for Downloading Configuration [32].

Chapter 7

Conceptual Data Model for Security Server

This chapter describes the entities and their attributes in the X-Road security server that are directly participating in configuration distribution. Figure 7.1 depicts the entities and their attributes, which are further detailed in tables 7.1 and 7.2.

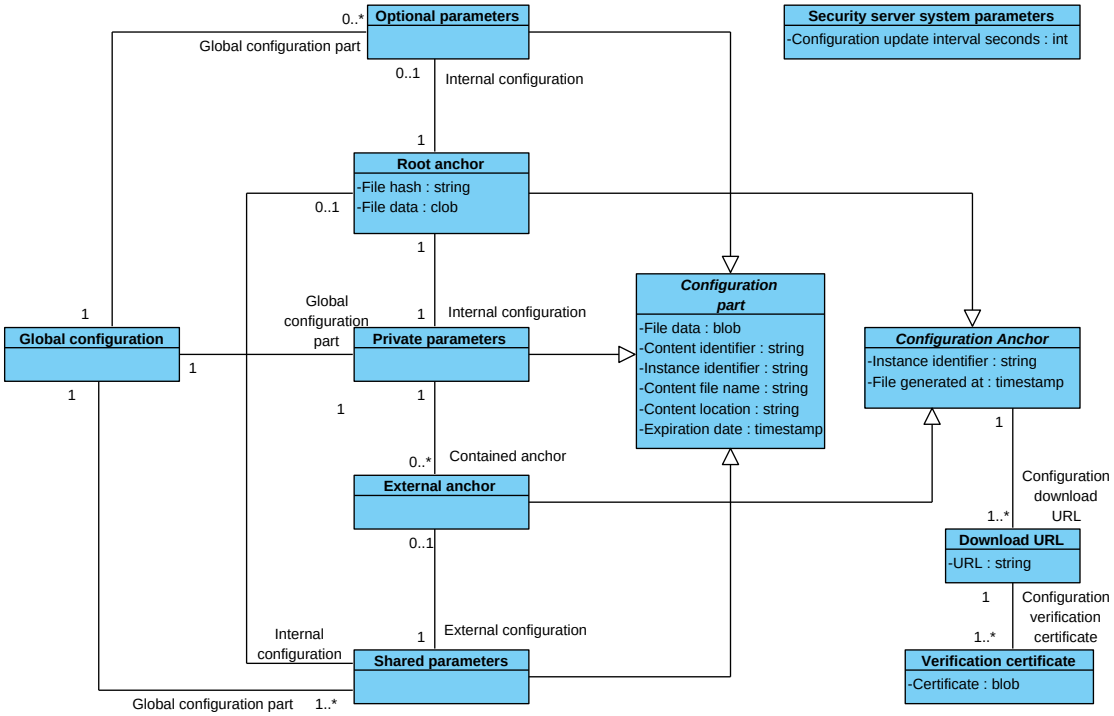


Figure 7.1: Conceptual data diagram for security server.

Table 7.1: Security server data model entities for configuration distribution.

Entity	Description
Global configuration	Global configuration is the generic name for all the configuration parts know to the system.
Configuration anchor	Set of information pointing to a configuration source distributing configuration.
Root anchor	Configuration anchor pointing to the configuration source that distributes internal configuration of the native X-Road system.
External anchor	Configuration anchor pointing to the configuration source that distributes external configuration of a federation partner.
Download URL	URL used for downloading configuration.
Verification certificate	Certificate used for verifying downloaded configuration.
Configuration part	Set of system parameters.
Private parameters	Configuration part containing the native X-Road system's internal information.
Shared parameters	Configuration part containing shared information of an X-Road system.
Optional parameters	Configuration part containing information of a specific X-Road system installation.
Security server system parameters	Security server inner parameters, not part of distributed configuration.

Table 7.2: Security server data model entity attributes for configuration distribution.

Entity	Attribute	Description
Global configuration	Private parameters	Private parameters belonging to the global configuration.
Global configuration	Shared parameters	Shared parameters belonging to the global configuration.
Global configuration	Optional parameters	Optional parameters belonging to the global configuration.

Configuration anchor	Instance identifier	Identifier of the X-Road instance the configuration anchor originates from.
Configuration anchor	File generated at	Generation date and time in ISO 8601 format [29] of the anchor file.
Configuration anchor	Download URL	Configuration download URL described in the anchor.
Root anchor	File hash	Hash of the root anchor file, computed using hash function SHA-224 [31].
Root anchor	File data	Root anchor file.
Root anchor	Private parameters	Private parameters of the native X-Road system, downloaded from the internal configuration source pointed by the root anchor.
Root anchor	Shared parameters	Shared parameters of the native X-Road system, downloaded from the internal configuration source pointed by the root anchor.
Root anchor	Optional parameters	Optional parameters of the native X-Road system, downloaded from the internal configuration source pointed by the root anchor.
External anchor	Shared parameters	Shared parameters of a federation partner's X-Road system, downloaded from the external configuration source pointed by the external anchor.
Download URL	URL	HTTP URL pointing to a configuration source.
Download URL	Verification certificate	Certificate that can be used to verify the configuration downloaded from the download URL.
Verification certificate	Certificate	Public key that can be used to verify the configuration downloaded from the download URL.
Configuration part	File data	Configuration part file.

Configuration part	Content identifier	Identifies the type of the configuration part. Predefined values are *"PRIVATE-PARAMETERS"* and *"SHARED-PARAMETERS"*. In addition, each X-Road installation using optional configuration parts must define additional content-identifiers for the optional parts.
Configuration part	Instance identifier	Identifier of the X-Road instance the configuration part originates from.
Configuration part	Content file name	Name of the configuration part file.
Configuration part	Content location	URL that can be used to download the configuration part file.
Configuration part	Expiration date	End of validity time of configuration. Contains date and UTC time in ISO 8601 format [29].
Private parameters	External anchor	Anchor contained in the private parameters, pointing to an external configuration source of a federation partner.
Security server system parameters	Configuration update interval seconds	System parameter defining the interval of configuration update.

Chapter 8

Central Server Use Case Model

8.1 Actors

The central server use case model includes the following actors.

User	Central server administrator - a person responsible for managing the central server.
Configuration client	System acting as a configuration client. Can either be a security server or a configuration proxy.

Relationships between actors and use cases are described in Figure 8.1.

8.2 General System Error Handling

Any system errors not defined in the use cases below, that may arise during user-system use cases (one actor is human) are handled as follows:

1. System logs error message.
2. System reports error to user.
3. System terminates use-case.

Any system errors not defined in the use cases below, that may arise during system use cases (no human actors) are handled as follows:

1. System logs error message.
2. System terminates use-case.

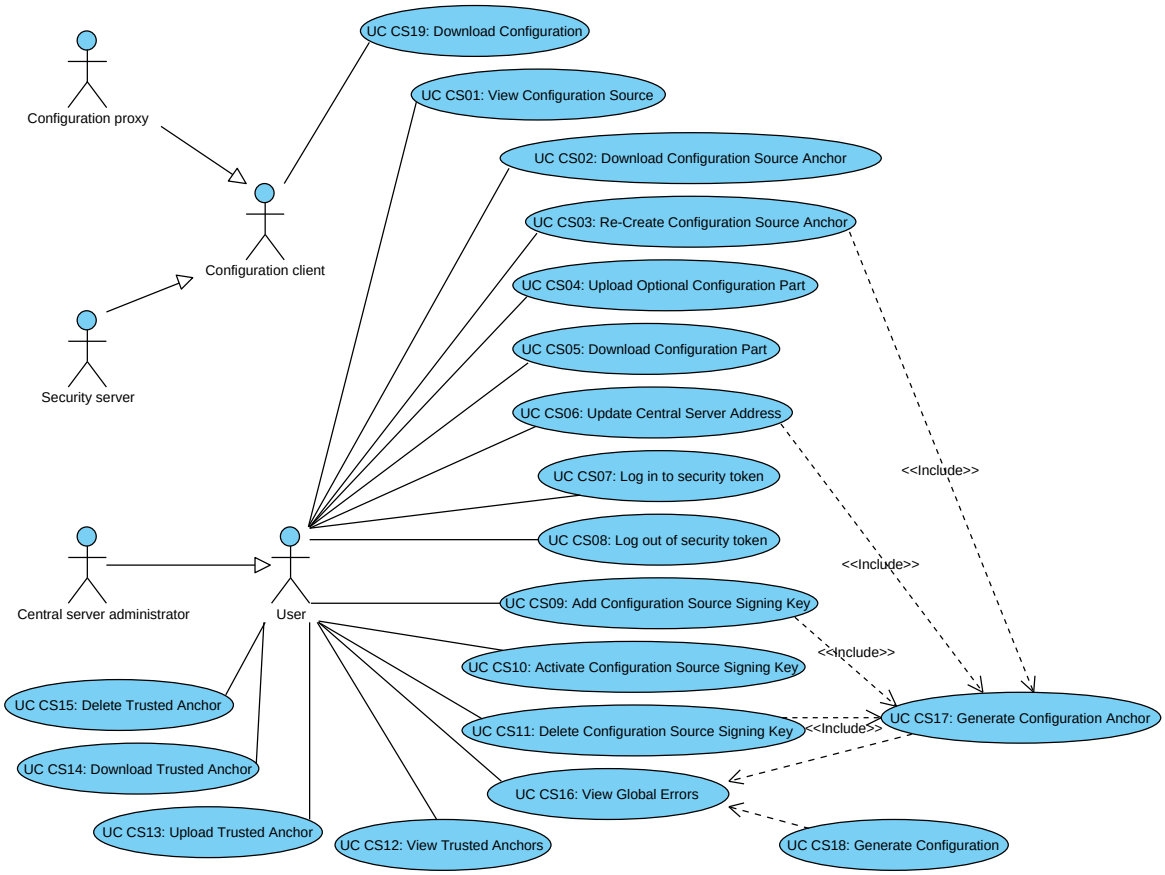


Figure 8.1: Use case diagram for central server.

8.3 User-System Use Cases

8.3.1 UC CS01: View Configuration Source

MAIN SUCCESS SCENARIO:

1. User selects to view configuration source.
2. System displays a configuration sources provided by the central server. Following information is displayed:
   - type (internal/external),
   - download URL,
   - configuration anchor hash and generation date and time (UTC),
   - list of signing keys that have a certificate associated with them. The key currently used to sign configuration is indicated as 'active',
   - list of configuration part files distributed by the source.

8.3.2 UC CS02: Download Configuration Source Anchor

PRECONDITION: Configuration anchor file is saved in the server's database.

MAIN SUCCESS SCENARIO:

1. User selects configuration source and selects to download configuration anchor.
2. System presents the configuration anchor file for downloading.
3. User saves the anchor file to computer's file system.

8.3.3 UC CS03: Re-Create Configuration Source Anchor

CONTEXT: Under normal system behavior, generation of anchor file by user is unnecessary, as the system generates the anchor automatically when needed. The re-creation allows the user to recover from system malfunctions.

PRECONDITION: Data needed to generate the anchor (instance identifier, download URL, certificate of a signing key) is saved in the server's database.

MAIN SUCCESS SCENARIO:

1. User selects configuration source and selects to generate configuration anchor.
2. System generates anchor (see UC CS17: Generate Configuration Anchor).

8.3.4 UC CS04: Upload Optional Configuration Part

PRECONDITION: Optional configuration part parameters are described in the system.

MAIN SUCCESS SCENARIO:

1. User selects an optional configuration part and selects to upload the configuration file.
2. User finds and selects the anchor from computer's file system and confirms the upload.
3. System validates the uploaded file.
4. System verifies the optional configuration part file already exists in the system's configuration and replaces the existing file with the uploaded one.

EXTENSIONS:

3a. Validation fails.

3a1. System reports validation faults to user and terminates use case.

4a. No previous file for this optional part exists in the system's database.

4a1. System saves the uploaded file.

8.3.5 UC CS05: Download Configuration Part

PRECONDITION: Configuration part file has been generated by the system or uploaded to the system.

MAIN SUCCESS SCENARIO:

1. User selects a configuration part and selects to download the configuration file.
2. System presents the configuration part for downloading.
3. User saves the configuration part file to computer's file system.

8.3.6 UC CS06: Update Central Server Address

MAIN SUCCESS SCENARIO:

1. User updates central server address.
2. System generates configuration anchor file (see UC CS17: Generate Configuration Anchor).
3. User downloads configuration source anchors (see UC CS02: Download Configuration Source Anchor).

8.3.7 UC CS07: Log In to Security Token

PRECONDITION: A security token is connected to the system and has not been logged in to.

MAIN SUCCESS SCENARIO:

1. User selects a key and selects to log in to the respective security token.
2. User enters PIN.
3. System verifies the PIN is correct and logs in to the token.

EXTENSIONS:

3a. PIN is incorrect:

3a1. System notifies user. Use case continues from step 2.

3b. Security token is locked (too many incorrect PIN entries):

3b1. System notifies user and terminates use case.

8.3.8 UC CS08: Log Out of Security Token

PRECONDITION: A security token is connected to the system and is logged in to.

MAIN SUCCESS SCENARIO:

1. User selects a key and selects to log out of the respective security token.
2. System logs out of the token.

8.3.9 UC CS09: Add Configuration Source Signing Key

PRECONDITION: A security token is connected to the system.

MAIN SUCCESS SCENARIO:

1. User selects a configuration source and selects to add new configuration source signing key.
2. System displays a list of available security tokens.
3. User selects a security token.
4. System generates a new configuration signing key on the selected token and corresponding self-signed certificate.
5. System saves the generated key information and certificate to database.
6. System verifies that the selected configuration source already has an active key.
7. System generates configuration anchor file (see UC CS17: Generate Configuration Anchor).

EXTENSIONS:

3a. Desired token is not on the list:

- 3a1. User connects the token to the system or terminates use case.

4a. Key generation fails: token PIN not entered:

- 4a1. System asks user for PIN
- 4a2. User enters PIN.
- 4a3. Use case continues from step 4.

4b. Generation of the self-signed certificate fails:

4b1. System deletes the generated key,

4b2. System reports failure to user, terminates use-case.

6a. The selected source does not have an active key.

6a1. System activates the generated key.

6a2. Use case continues from step 7.

8.3.10 UC CS10: Activate Configuration Source Signing Key

PRECONDITION: An inactive signing key is associated with a configuration source.

MAIN SUCCESS SCENARIO:

1. User selects a configuration source and selects to activate an inactive signing key.
2. System prompts for confirmation.
3. User confirms.
4. System starts using the activated key for signing configuration.

EXTENSIONS:

3a. User cancels key activation:

3a1. System terminates use case.

8.3.11 UC CS11: Delete Configuration Source Signing Key

PRECONDITION: An inactive signing key is associated with a configuration source.

MAIN SUCCESS SCENARIO:

1. User selects a configuration source and selects to delete a configuration source signing key.
2. System prompts for confirmation.
3. User confirms.
4. System deletes the selected configuration signing key from the security token and the corresponding certificate and key ID from the database.
5. System generates configuration anchor file (see UC CS17: Generate Configuration Anchor).

EXTENSIONS:

3a. User cancels key deletion:

3a1. System terminates use case.

8.3.12 UC CS12: View Trusted Anchors

MAIN SUCCESS SCENARIO:

1. User selects to view trusted anchors.
2. System displays a list of trusted anchors known to the central server. For each anchor, following information is displayed:
   - instance identifier,
   - anchor file hash and
   - generation date and time (UTC).

8.3.13 UC CS13: Upload Trusted Anchor

TRIGGERS: Federation of an X-Road instance; federation partner sends updated configuration anchor.

PRECONDITIONS: User has received a configuration anchor from a configuration provider and validated the integrity of the anchor.

MAIN SUCCESS SCENARIO:

1. User selects to upload configuration anchor.
2. User finds and selects the anchor from computer's file system.
3. System displays anchor description and asks user to confirm the import.
4. User confirms.
5. System imports the anchor file.
6. System downloads signed configuration from the source defined by the uploaded anchor and validates the configuration.
7. System verifies that an anchor with the same instance identifier as the uploaded one exists in the system's private parameters and replaces the existing anchor with the uploaded one.

EXTENSIONS:

4a. User cancels the import:

4a1. System terminates use case.

7a. No anchor with the same instance identifier as the uploaded one exists in the system's private parameters

7a1. System adds the uploaded anchor to system's private parameters.

8.3.14 UC CS14: Download Trusted Anchor

PRECONDITIONS: A trusted anchor has been uploaded to the system.

MAIN SUCCESS SCENARIO:

1. User selects a trusted anchor and selects to download the anchor file.
2. System presents the anchor file for downloading.
3. User saves the anchor file to computer's file system.

8.3.15 UC CS15: Delete Trusted Anchor

TRIGGERS: A federation relationship is terminated.

PRECONDITIONS: A trusted anchor has been uploaded to the system.

MAIN SUCCESS SCENARIO:

1. User selects a configuration anchor and selects to delete the anchor.
2. System prompts for confirmation.
3. User confirms the deletion.
4. System deletes the selected configuration anchor.

EXTENSIONS:

3a. User cancels the deletion:

3a1. System terminates use case.

8.3.16 UC CS16: View Global Error Messages

PRECONDITIONS: An error message has been saved.

MAIN SUCCESS SCENARIO:

1. System displays saved error messages.
2. User acts on the displayed message.

8.4 System Use Cases

8.4.1 UC CS17: Generate Configuration Anchor

TRIGGERS: Configuration source download URL is changed or a signing key is added or deleted.

MAIN SUCCESS SCENARIO:

1. System verifies that information needed to generate the anchor (instance identifier, central server address, at least one signing key with corresponding certificate) is saved in the system's database.
2. System generates the anchor file and calculates the file hash.
3. System saves the anchor file, file hash and file generation time to database.

EXTENSIONS:

*. Anchor generation fails:

- *1. System logs errors.
- *2. System saves error messages for displaying to user (see UC CS16: View Global Error Messages).
- *3. System terminates use case.

8.4.2 UC CS18: Generate Configuration

TRIGGER: Timer (defined by system parameter *configuration generation interval seconds*, implemented by *cron* daemon).

MAIN SUCCESS SCENARIO:

1. System generates configuration part files for private and shared parameters, saves the files to database and adds the respective file information (content type, content transfer encoding, content file name, content identifier, content location, hash algorithm id, file digest) to configuration directory.
2. Systems adds information (content type, content transfer encoding, content file name, content identifier, content location, hash algorithm id, file digest) about optional configuration parts to configuration directory.
3. System adds parameters (instance identifier, expire date) to the configuration directory.
4. System signs the configuration directory (signature information includes content type, content transfer encoding, signature value, signature algorithm id, verification certificate hash, hash algorithm id).
5. System makes the signed directory and configuration part files available to configuration clients.

EXTENSIONS:

*. Configuration generation fails:

*1. System logs errors.

*2. System saves error messages for displaying to user (see UC CS16: View Global Error Messages).

*3. System terminates use case.

8.4.3 UC CS19: Download Configuration

TRIGGER: Configuration download request from a configuration client.

MAIN SUCCESS SCENARIO:

1. Configuration client requests to download configuration.
2. System responds with the requested files.

EXTENSIONS:

2a. Request cannot be served.

2a1. System responds with error message.

Chapter 9

Security Server Use Case Model

9.1 Actors

The security server use case model includes one actor:

User	Security server administrator - a person responsible for managing the security server.

Relationships between the actor and use cases are described in Figure 9.1.

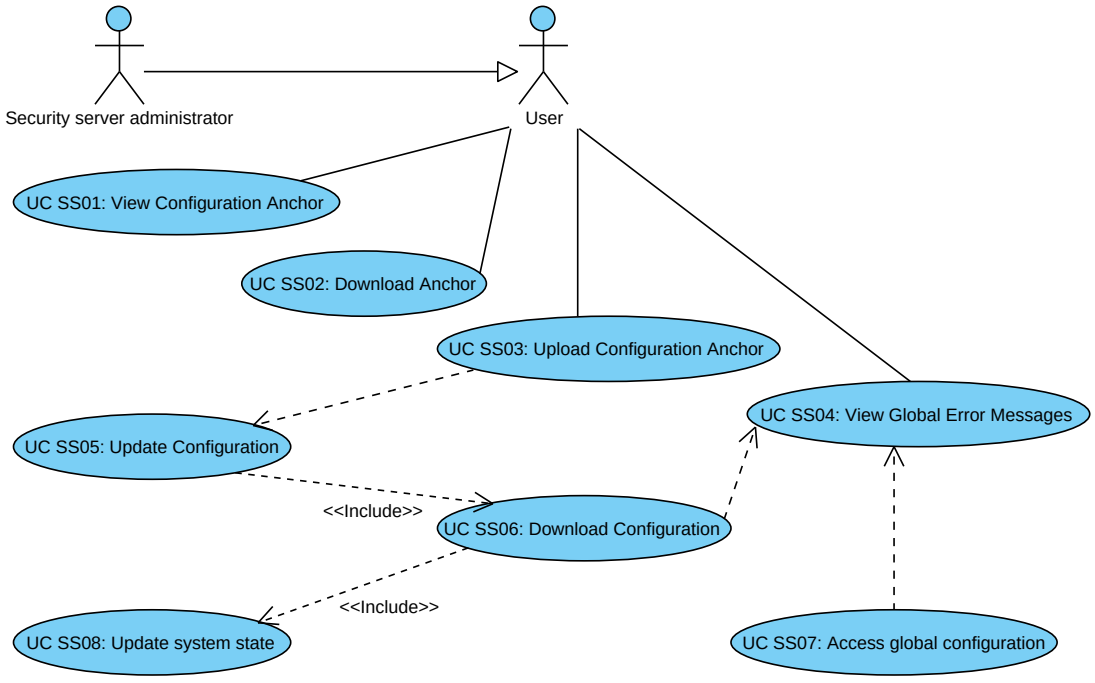


Figure 9.1: Use case diagram for security server.

9.2 General System Error Handling

Any system errors not defined in the use cases below, that may arise during user-system use cases (one actor is human) are handled as follows:

1. System logs error message.
2. System reports error to user.
3. System terminates use-case.

Any system errors not defined in the use cases below, that may arise during system use cases (no human actors) are handled as follows:

1. System logs error message.
2. System terminates use-case.

9.3 User-System Use Cases

9.3.1 UC SS01: View Configuration Anchor

MAIN SUCCESS SCENARIO:

1. User selects to view configuration anchor.
2. System displays the following information:
    - anchor file hash and
    - anchor generation time.

9.3.2 UC SS02: Download Configuration Anchor

PRECONDITIONS: A trusted anchor has been uploaded to the system.
MAIN SUCCESS SCENARIO:

1. User selects to download the anchor file.
2. System presents the anchor file for downloading.
3. User saves the anchor file to computer's file system.

9.3.3 UC SS03: Upload Configuration Anchor

TRIGGERS: Security server initialization; modification of the address or certificate set of the configuration source containing private parameters.

PRECONDITIONS: User has received the anchor form its internal configuration provider and validated the integrity of the anchor.

MAIN SUCCESS SCENARIO:

1. User selects to upload configuration anchor.
2. User finds and selects the anchor from computer's file system.
3. System displays anchor description and asks user to confirm the import.
4. User confirms.
5. System imports the anchor file.
6. System downloads configuration (see UC SS05: Update Configuration) from the source pointed by the uploaded anchor.
7. System starts using the uploaded anchor and downloaded configuration.

EXTENSIONS:

4a. User cancels the import:

4a1. System terminates use case.

9.3.4 UC SS04: View Global Error Messages

PRECONDITIONS: An error message has been saved.

MAIN SUCCESS SCENARIO:

1. System displays saved error messages.
2. User acts on the displayed message.

9.4 System Use Cases

9.4.1 UC SS05: Update Configuration

TRIGGERS: Configuration anchor import; timer (defined by system parameter *configuration update interval seconds*).

PRECONDITIONS: configuration anchor is uploaded into the system.

MAIN SUCCESS SCENARIO:

1. System downloads configuration (see UC SS06: Download Configuration) from the source pointed by the root anchor.
2. System finds configuration anchors pointing to external configuration sources from the private parameters part of the internal configuration and downloads configuration (see UC SS06: Download Configuration) from each source pointed by the anchors.
3. System starts using the downloaded configuration.

9.4.2 UC SS06: Download Configuration

PRECONDITIONS: configuration anchor exists.

MAIN SUCCESS SCENARIO:

1. System downloads signed configuration directory by making a HTTP GET request to the download URL specified by the configuration anchor.
2. System verifies the signature with one of the certificates found in the configuration anchor.
3. System verifies that the downloaded configuration is not expired (compares *Expire-date* header value of the configuration directory to current date) and saves the expiry date.
4. System verifies that the set of configuration part hash values contained in the downloaded directory is identical to the set of hash values of the configuration part files existing in the system.
5. System updates state (see UC SS08: Update System State).

EXTENSIONS:

1a. Download fails.

1a1. System saves error messages for displaying to user (see UC SS04: View Global Error Messages).

2a. Verification fails.

2a1. System saves error messages for displaying to user (see UC SS04: View Global Error Messages).

3a. Configuration is expired.

3a1. System saves error messages for displaying to user (see UC SS04: View Global Error Messages).

4a. One or more configuration parts present in the system are missing from the downloaded directory.

4a1. System deletes configuration files that are missing from the downloaded directory.

4a2. Use case continues from step 5.

4b. One or more configuration parts present in the downloaded directory are missing from the system.

4b1. System downloads the missing configuration files by making a HTTP GET request to the URL specified by the *Content-location* header of the directory part corresponding to the missing file.

4b2. System stores the downloaded configuration files.

4b3. Use case continues from step 5.

4c. One or more configuration part hashes in the downloaded directory differ from the hashes of the files present in the system.

4c1. System downloads the changed configuration files by making a HTTP GET request to the URL specified by the *Content-location* header of the directory part corresponding to the changed file.

4c2. System replaces the existing configuration file(s) with the downloaded one(s).

4c3. Use case continues from step 5.

9.4.3 UC SS07: Access Global Configuration

TRIGGER: System needs information from global configuration.

PRECONDITIONS: Configuration directory exists.

MAIN SUCCESS SCENARIO:

1. System verifies that configuration is not expired (compares *Expire-date* header value of the configuration directory to current date).
2. System accesses configuration file and gets data.

EXTENSIONS:

1a. Configuration is expired.

1a1. System saves error messages for displaying to user (see UC SS04: View Global Error Messages).

1a3. System disables the following functionality:

- X-Road message mediation (refuses any incoming messages);
- certificate import (no certificates can be imported to the system);
- OCSP response querying;
- modification of time-stamping services;
- time-stamping of X-Road messages;
- (in case of X-Road version 5.5) activation and promotion of the version 6 components.

9.4.4 UC SS08: Update System State

MAIN SUCCESS SCENARIO:

1. System updates authentication certificate states.
2. System updates server client states.

Chapter 10

User Interfaces

10.1 User Roles and Privileges

User actions in graphical user interfaces of the X-Road system components are tied to user groups. User groups are assigned with a set of privileges, each privilege granting access to a user action.

The management of user groups and privileges relies on the operation system facilities, allowing

- use of different user management systems (local, LDAP, etc),
- standardized and organization-specific creation and management of user rights,
- better auditability through transparency and uniformity of user management. (Margus Freudenthal, personal communication, August 2013)

The following rules apply to user roles:

- one user may belong to many user groups;
- one user group may contain many users;
- the set of user privileges comprises the union of privileges assigned to each user groups the user belongs to.

The user logged in to a GUI only sees the menu items and buttons that are tied to the user's privileges.

10.1.1 Central Server User Roles, Groups and Privileges

Table 10.1 describes the central server user roles and user groups involved in configuration management.

Table 10.1: Central server user roles and user groups.

User role	User group identifier	Description
Security officer	sdsb-security-officer	Responsible for the application of the security policy and security requirements.
System administrator	sdsb-system-administrator	Responsible for the installation, configuration, and everyday maintenance of the central server.
Registration officer	sdsb-registration-officer	Responsible for handling the information about X-Road members.

Table 10.2describes the user action (detailed in use cases) privileges given to user roles.

Table 10.2: Central server user roles and user action privileges.

User role	User action privilege
Security officer	UC CS01: View Configuration Source
Security officer	UC CS02: Download Configuration Source Anchor
Security officer	UC CS03: Re-Create Configuration Source Anchor
Security officer	UC CS04: Upload Optional Configuration Part
Security officer	UC CS05: Download Configuration Part
Security officer	UC CS06: Update Central Server Address
Security officer	UC CS07: Log In to Security Token
Security officer	UC CS08: Log Out of Security Token
Security officer	UC CS09: Add Configuration Source Signing Key
Security officer	UC CS10: Activate Configuration Source Signing Key
Security officer	UC CS11: Delete Configuration Source Signing Key
Security officer	UC CS12: View Trusted Anchors
Security officer	UC CS13: Upload Trusted Anchor
Security officer	UC CS14: Download Trusted Anchor
Security officer	UC CS15: Delete Trusted Anchor
Security officer	UC CS16: View Global Error Messages
System administrator	UC CS01: View Configuration Source
System administrator	UC CS05: Download Configuration Part
System administrator	UC CS07: Log In to Security Token
System administrator	UC CS08: Log Out of Security Token
System administrator	UC CS12: View Trusted Anchors

System administrator	UC CS14: Download Trusted Anchor
System administrator	UC CS16: View Global Error Messages
Registration officer	UC CS01: View Configuration Source
Registration officer	UC CS04: Upload Optional Configuration Part*
Registration officer	UC CS12: View Trusted Anchors
Registration officer	UC CS16: View Global Error Messages

* Applies to X-Road system version 5.5.

10.2 Central Server User Interface Specification

This section details the additions to the central server graphical user interface to allow the users of the system to manage configuration.

10.2.1 Configuration Management View

Menu item **Configuration Management** is added to the main menu of the central server user interface, under the **Management** section.

The configuration management view opened by the respective main menu item is divided into sub-views, each of which is accessible by a tab button.

- The **Internal Configuration** tab button
  - opens the sub-view for managing internal configuration;
  - implements use case UC CS01: View Configuration Source.
- The **External Configuration** tab button
  - opens the sub-view for managing external configuration;
  - implements use case UC CS01: View Configuration Source.
- The **Trusted Anchors** tab button
  - opens the sub-view for managing trusted anchors;
  - implements use case UC CS12: View Trusted Anchors.

Details of the sub-views are described in the following sections.

10.2.2 Internal/External Configuration Sub-View

The design of the sub-view for managing internal and external configuration is depicted in Figure 10.1 (for the screenshot of the actual GUI view see A.1). The internal and external configuration sub-views are visually and functionally identical (except where specifically mentioned in the specification below). Internal/external configuration sub-view consists of four visually distinguishable sections – **Anchor**, **Download URL**, **Signing Keys** and **Configuration Parts**.

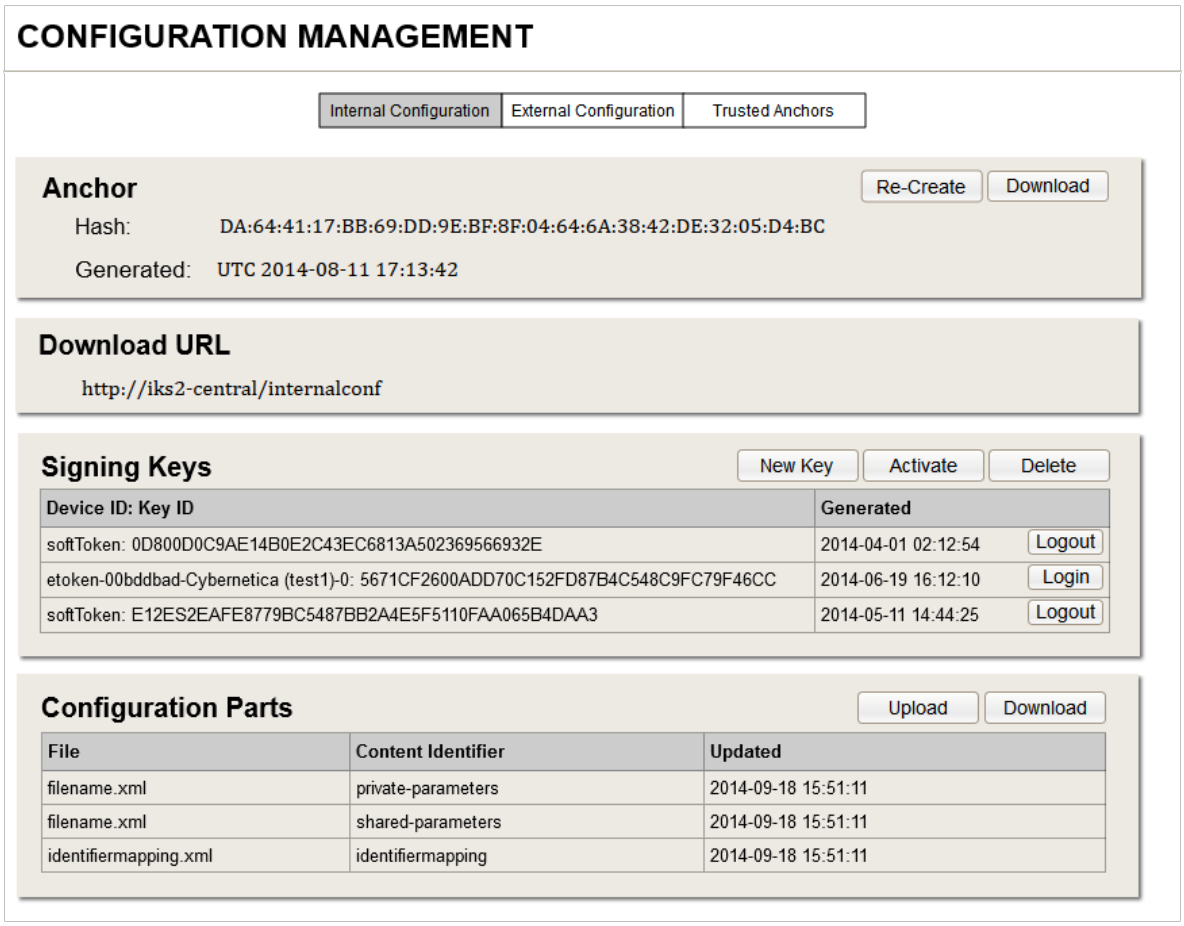


Figure 10.1: User interface design draft for central server configuration sources sub-view.

Section: Anchor

The anchor section contains information and functionality related to the configuration anchor of the configuration source distributing this configuration.

Following information is displayed.

- If anchor file does not exist, message "Anchor file not found." is displayed.
- If anchor file has been generated, following label fields are displayed:

  - Hash: file hash value (SHA-224) of the configuration anchor.
  - Generated: UTC generation time of the configuration anchor file, formatted as "UTC YYYY-MM-DD hh:mm:ss".

The section contains following buttons.

- Re-Create: implements use case UC CS03: Re-Create Configuration Source Anchor.
- Download:
  - implements use case UC CS02: Download Configuration Source Anchor;
  - file name of the downloaded anchor is formatted as "configuration_anchor_<instance_identifier>_<configuration_type>_ <generated_at>.xml" (e.g. "configuration_anchor_AA_internal_UTC_2014-11-28_15_56_50.xml").

Section: Download URL

The download URL section displays the download URL of the configuration source distributing this configuration, formatted as "<central_server_URL>/<configuration_source_URI>".

Section: Signing Keys

This section contains information and functionality related to the signing keys of this configuration.

The information about the keys is displayed as a table, each row representing one key. The table consists of following columns.

- Device ID: Key ID: displays the friendly name of the security token holding the key and the identifier of the key, separated by colon.
- Generated: displays the generation time of the key as local time, formatted as "YYYY-MM-DD hh:mm:ss".

The following rules apply to the rows.

- Active key is displayed in bold font.
- Key inaccessible to the system (e.g. the hardware token carrying the key is unattached to the system) is grayed out.

The section contains following buttons.

- New Key: implements use case UC CS09: Add Configuration Source Signing Key.
- (For each row in table) Login/Logout:
  - if the security token carrying the key is not logged in to, Login button is displayed - implements use case UC CS07: Log In to Security Token;
  - if the security device carrying the key is logged in to, Logout button is displayed - implements use case UC CS08: Log Out of Security Token.
- Activate:
  - enabled, if an inactive key row is selected, else grayed out;
  - implements use case UC CS10: Activate Configuration Source Signing Key.
- Delete:
  - enabled, if an inactive key row is selected, else grayed out;
  - implements use case UC CS11: Delete Configuration Source Signing Key.

Section: Configuration Parts

This section contains information and functionality related to the configuration parts of this configuration.

The information about the configuration parts is displayed as a table, each row representing one part. The table consists of following columns.

- File: displays the file name of the configuration part file.
- Content Identifier: displays the content identifier of the configuration part.
- Updated: displays the generation time (in case of shared parameters or private parameters) or upload time (in case of optional parameters) of the configuration part file as local time, formatted as “YYYY-MM-DD hh:mm:ss”.

The section contains following buttons.

- Upload:
  - only displayed in the internal configuration sub-view;
  - enabled, if an optional configuration part row is selected, else grayed out;
  - implements use case UC CS04: Upload Optional Configuration Part.

- Download:
  - enabled, if a row is selected, else grayed out;
  - implements use case UC CS05: Download Configuration Part;
  - file name of the downloaded configuration part is formatted as "<file_name>_<updated>" (e.g. "identifiermapping_2014-09-18_15_51_11.xml").

10.2.3 Trusted Anchors Sub-View

The design of the sub-view for managing trusted anchors is depicted in Figure 10.2 (for the screenshot of the actual GUI view see A.2). The view holds information about configuration anchors pointing to configuration sources distributing external configuration of X-Road systems federated with the native system.

For each federation partner, an anchor section is displayed. If no trusted anchors have been uploaded to the system, message: "No trusted anchors found" is displayed in place of the anchor sections.

Upon selecting the trusted anchors sub-view, the following general button is displayed.

- Upload Anchor: implements use case 8.3.13.

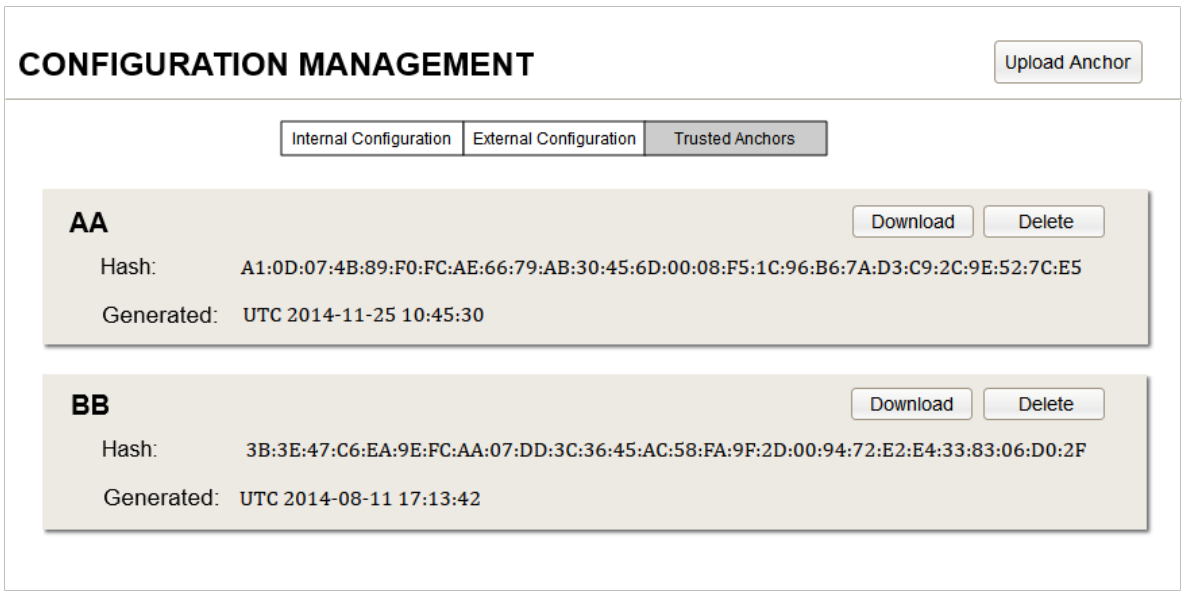


Figure 10.2: User interface design draft for central server trusted anchors sub-view.

Section for each trusted anchor

Following information is displayed.

- Instance identifier of the X-Road system the configuration obtainable via the anchor originates from is displayed as the section header.
- Hash: file hash value (SHA-224) of the configuration anchor.
- Generated: UTC generation time of the configuration anchor file, formatted as “UTC YYYY-MM-DD hh:mm:ss”.

The section contains following buttons.

- Download:
  - implements use case 8.3.14.
  - file name of the downloaded trusted anchor is formatted as “configuration_anchor_<instance_identifier>_<configuration_type>_ <generated_at>.xml” (e.g. “configuration_anchor_BB_external_UTC_2014-12-18_11_08_21”).
- Delete: implements use case 8.3.15.

10.3 Security Server User Interface Specification

10.3.1 Security Server User Roles, Groups and Privileges

Table 10.3 describes the security server user roles and user groups involved in configuration management.

Table 10.3: Security server user roles and user groups.

User role	User group identifier	Description
Security officer	sdsb-security-officer	Responsible for the application of the security policy and security requirements.
System administrator	sdsb-system-administrator	Responsible for the installation, configuration, and everyday maintenance of the security server.

Table Security Server User Roles, Groups and Privileges describes the user action (detailed in use cases) privileges given to user roles.

Table 10.4: Security server user roles and user action privileges

User role	User action privilege
Security officer	UC SS01: View Configuration Anchor
Security officer	UC SS02: Download Configuration Anchor
Security officer	UC SS03: Upload Configuration Anchor
Security officer	UC SS04: View Global Error Messages
System administrator	UC SS01: View Configuration Anchor
System administrator	UC SS02: Download Configuration Anchor
System administrator	UC SS04: View Global Error Messages

10.3.2 Configuration Anchor Section in the System Parameters View

This section details the additions to the security server graphical user interface to allow the users of the system to manage root configuration anchor.

Section **Configuration Anchor** is added to the **System Parameters** view. The System Parameters menu item in the main menu of the security server's graphical user interface implements use case UC SS01: View Configuration Anchor (for the screenshot of the actual GUI view see A.3). The design of the section is depicted in Figure 10.1.

Figure 10.3: User interface design draft for security server configuration anchor section.

The configuration anchor section displays the following information.

- Hash: file hash value (SHA-224) of the root configuration anchor.
- Generated: UTC generation time of the configuration anchor file, formatted as "UTC YYYY-MM-DD hh:mm:ss".

The section contains following buttons.

- Upload: implements use case UC SS03: Upload Configuration Anchor.
- Download:

- implements use case UC SS02: Download Configuration Anchor;
- file name of the downloaded anchor is formatted as "configuration_anchor_<generated_at>.xml" (e.g. "configuration_anchor_UTC_2014-12-15_16_13_06.xml").

10.4 Global Error Messages

Global error messages inform the user of configuration or system errors that render the system incapable of carrying out its primary function.

The primary function for

- central server is to distribute configuration;
- security server is to mediate X-Road messages.

Global error messages shown in graphical user interfaces implement use cases UC CS16: View Global Error Messagesand UC SS04: View Global Error Messages. Screenshot showing global error messages in central server GUI can be found in Appendix A A.4.

Summary

The project “Development of X-Road system supporting cross-border services” resulted in an X-Road system capable of entering into a federation relationship with another X-Road system and supporting cross-border data exchange between organizations belonging to the federated systems. Establishing the federation relationship is carried out at the X-Road governing agency’s level. The technical capability for cross-border data exchange is achieved through configuration management. The configuration distributed by the governing agency to the security servers used by organizations carries all the information needed for cross-border data exchange. The functionality for supporting cross-border e-services was developed using the system analysis presented in this thesis.

The development of the system was carried out using RUP methodology. The system analysis artifacts were created in the inception and elaboration phases.

The system concepts were described in the inception phase, using conceptual model to define the main entities and relations, and business use case model to describe the most significant processes in the federated X-Road systems. In the elaboration phase, the functionality of the system components was detailed in use case models and, in parallel, data objects were described in conceptual data models. System component user interface specifications were created to describe the design and functionality of user interfaces. The X-Road user interface user roles were described. For each role, a set of user action privileges for the added functionality was defined. The artifacts created in the elaboration phase were used by programmers to implement the system and by testers for testing the implemented functionality.

The used methodology and conducted analysis (in conjunction with the artifacts created by the system architect) proved to be suitable and sufficient for development of X-Road system supporting cross-border services.

Kokkuvõte

Piiriüleseid teenuseid toetav X-tee süsteem on võimeline astuma föderatsioonisuhtesse teise X-tee süsteemiga ning võimaldab piiriülest andmevahetust födereerunud süsteemidesse kuuluvate organisatsioonide vahel. Föderatsioonisuhte loomine toimub X-tee keskuse tasemel, tehniline võimekus piiriüleseks andmevahetuseks luuakse konfiguratsioonihalduse kaudu – andmeid vahendavatesse organisatsioonidesse paigaldatud turvaserverid saavad kogu piiriüleseks andmevahetuseks vajaliku informatsiooni keskuse poolt jagatava konfiguratsiooniga. Piiriüleseid teenuseid võimaldav funktsionaalsus lisati X-tee süsteemile käesolevas töös kirjeldatud süsteemianalüüsi tulemite põhjal.

Süsteemi arendus toimus RUP metoodikat kasutades, analüüsitulemid loodi algatus- ja detailimisfaasi käigus.

Süsteemi kontseptsiooni kirjeldamiseks loodi arenduse algatusfaasis süsteemi kontseptuaalne mudel, milles defineeriti süsteemi põhiolemid ja olemitevahelised seosed ning talitlusmallimudel kirjeldamaks olulisemaid talitlusprotsesse födereerunud X-teede süsteemis. Detailimisfaasis kirjeldati süsteemi komponentidele arenduse käigus lisanduv funktsionaalsus kasutusmallimudelites ja paralleelselt modelleeriti kontseptuaalses andmemudelis tähtsaimad andmeobjektid. Kasutajaliideste disaini ja funktsionaalsuse kirjeldamiseks loodi süsteemi komponentide kasutajaliideste spetsifikatsioonid. Kirjeldati X-tee süsteemi komponentide kasutajaliideste kasutajarollid ning igale rollile defineeriti privileegide komplekt arenduse käigus lisanduva funktsionaalsuse kasutamiseks. Detailimisfaasi tulemeid kasutasid programmeerijad süsteemi realiseerimisel ja testijad süsteemi funktsionaaluse testimisel.

Kasutatud metoodika ja analüüsi käigus loodud tehised (koostöös süsteemiarhitekti poolt loodud tehistega) olid sobivad ja piisavad piiriüleseid teenuseid toetava X-tee süsteemi arendamiseks.

Bibliography

[1] Eesti infoühiskonna arengukava 2020. Available online at http://infoyhiskond.eesti.ee/files/Infoyhiskonna_arengukava_2020_f.pdf, 2013. Accessed 04.01.2015.

[2] Piiriüleseid teenuseid toetava X-tee arendamine. Available online at https://riigihanked.riik.ee/register/hange/150193. Accessed: 03.01.2015.

[3] Kansallinen palveluväylä – konsepti, tavoitteet ja ratkaisumalli. Available online at http://www.vm.fi/vm/fi/04_julkaisut_ja_asiakirjat/03_muut_asiakirjat/20130516Kansal/name.jsp, 2013. Accessed 04.01.2015.

[4] Philippe Kruchten Per Kroll˙ *The Rational Unified Process Made Easy: A Practitioners Guide to the RUP*. Addison-Wesley Professional, Boston, 2003.

[5] Integreeritud juhtimissüsteem: Tarkvaraarendus. Organizational regulation Y-415-1, 2013.

[6] Martin Fowler. *UML Distilled, Third Edition*. Addison-Wesley Professional, Boston, 2003.

[7] Unified Modeling Language (UML) Resource Page. Available online at http://www.uml.org/. Accessed: 13.01.2015.

[8] Visual Paradigm. Available online at http://www.visual-paradigm.com/. Accessed: 13.01.2015.

[9] Alistair Cockburn. *Writing Effective Use Cases, 1st*. Addison-Wesley Longman Publishing Co., Boston, 2000.

[10] Pencil Project. Available online at http://pencil.evolus.vn/. Accessed: 13.01.2015.

[11] X-Road. Available online at http://cyber.ee/en/e-government/x-road/. Accessed: 02.01.2015.

[12] *Public Information Act*. Riigi Teataja, 2000.

[13] Data Exchange Layer X-Road. Available online at https://www.ria.ee/x-road/. Accessed 02.01.2015.

[14] Secure Distributed Service Bus. Available online at http://www.eliko.ee/secure-distributed-service-bus. Accessed: 13.01.2015.

[15] Foundation architectures: X-road. Technical document Y-791-1, Cybernetica AS, 2014.

[16] Simple Object Access Protocol (SOAP) 1.1. W3C Note, May 2000.

[17] Web Services Description Language (WSDL) 1.1. W3C Note, March 2001.

[18] The MIME Multipart/Related Content-type. Request for Comments 2387, Internet Engineering Task Force, August 1998.

[19] Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile. Standard 5280, Internet Engineering Task Force, may 2008.

[20] The Transport Layer Security (TLS) Protocol Version 1.2. Request for Comments 5246, Internet Engineering Task Force, August 2008.

[21] Internet X.509 Public Key Infrastructure. Time-Stamp Protocol (TSP). Request for Comments 3161, Internet Engineering Task Force, August 2001.

[22] X.509 Internet Public Key Infrastructure. Online Certificate Status Protocol - OCSP. Request for Comments 6960, Internet Engineering Task Force, June 2013.

[23] X-tee2 visioon. Technical Document Y-743-2, Cybernetica AS, 2012.

[24] Electronic Signatures and Infrastructures (ESI); XML Advanced Electronic Signatures (XAdES). Technical Specification TS 101 903, ETSI ESI, December 2010.

[25] Electronic Signatures and Infrastructures (ESI); Associated Signature Containers (ASiC), Version 1.3.1. Technical Specification TS 102 918, ETSI ESI, June 2013.

[26] Margus Freudenthal. Using Batch Hashing for Signing and Time-Stamping. Research Report T-4-20, Cybernetica AS, 2013.

[27] Soome rajab andmevahetuskihi Eesti X-tee eeskujul. Available online at https://www.ria.ee/soome-rajab-andmevahetuskihi/, September 2013. Accessed: 04.01.2015.

[28] Piiriüleseid teenuseid toetava X-tee arendamine. Tender M-43-91, July 2014.

[29] Data Elements and Interchange Formats – Information Interchange – Representation of Dates and Times. ISO 8601:2004, International Organization for Standardization, 2004.

[30] XML Signature Syntax and Processing Version 2.0. W3C recommendation, apr 2013.

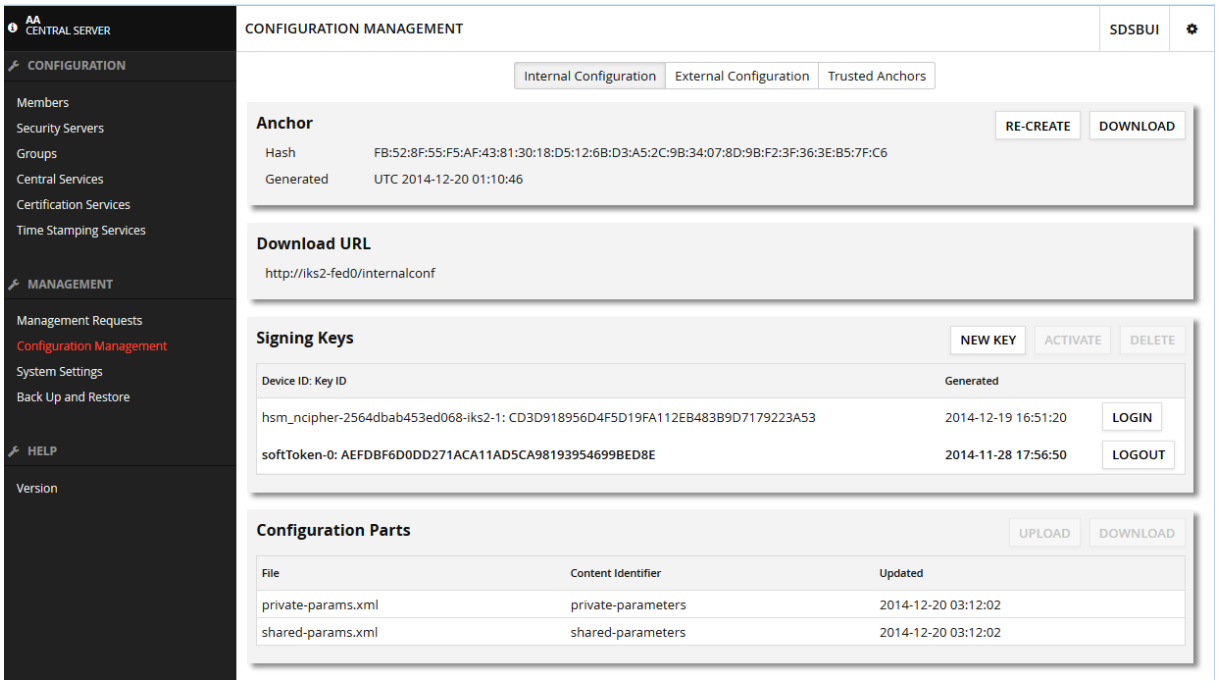
[31] A 224-bit one-way hash function: SHA-224. Informational 3874, Internet Engineering Task Force, September 2004.

[32] Protocol for Downloading Configuration. Technical document Y-743-15, Cybernetica, 2014.

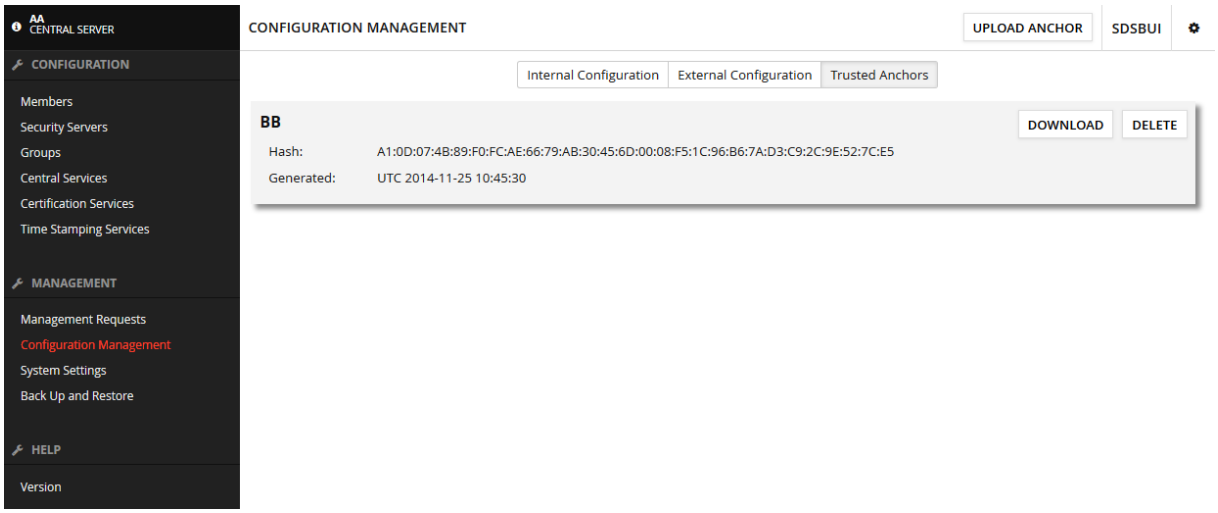
Appendix A

Screenshots of Graphical User Interfaces

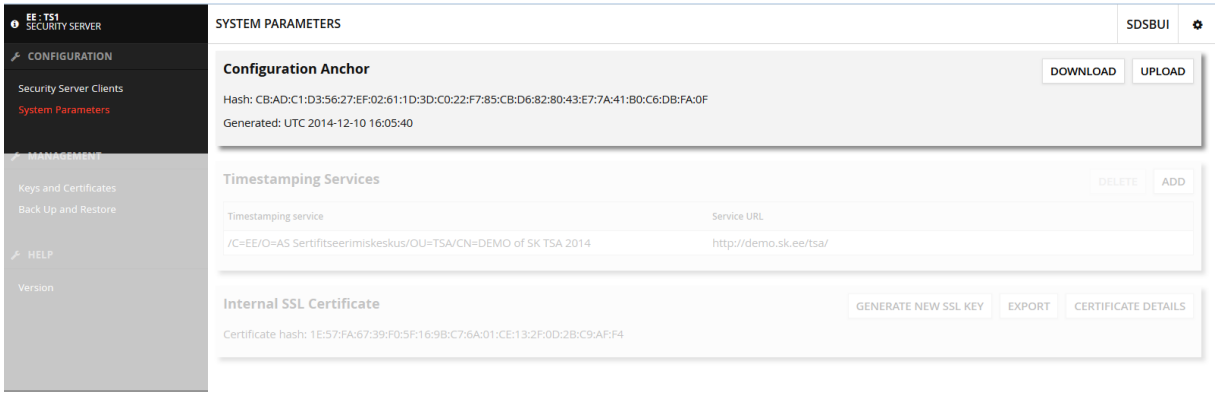
A.1 Central Server Configuration Management: Internal Configuration



A.2 Central Server Configuration Management: Trusted Anchors



A.3 Security Server System Parameters: Configuration Anchor



A.4 Central Server: Global Error Messages