Smart-ID SecureZone Security Target

Technical document
Version 2.7.0
September 21, 2018
111 pages

Contents

List of Figures

List of Tables

1 Introduction

1.1 Objectives and Scope of the Document

This document is the Security Target (ST) document for the Smart-ID SecureZone. The ST defines the Target of Evaluation and describes the security problem with the terms of Common Criteria.

1.2 Intended Audience

TOE users, developers, evaluators and certifiers.

1.3 Related Documents

1.3.1 Normative references

[1] *Common Criteria for Information Technology Security Evaluation. Part 1: Introduction and general model*, Apr. 2017. [Online]. Available: https://www.commoncriteriaportal.org/files/ccfiles/CCPART1V3.1R5.pdf.

[2] *Common Criteria for Information Technology Security Evaluation. Part 2: Functional security components*, Apr. 2017. [Online]. Available: https://www.commoncriteriaportal.org/files/ccfiles/CCPART2V3.1R5.pdf.

[3] *Common Criteria for Information Technology Security Evaluation. Part 3: Assurance security components*, Apr. 2017. [Online]. Available: https://www.commoncriteriaportal.org/files/ccfiles/CCPART3V3.1R5.pdf.

[4] *Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC*, Aug. 2014. [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=uriserv%3AOJ.L_.2014.257.01.0073.01.ENG.

1.3.2 Other references

[5] A. Buldas, A. Kalu, P. Laud, and M. Oruaas, "Server-Supported RSA Signatures for Mobile Devices", in *Computer Security – ESORICS 2017: 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part I*, S. N. Foley, D. Gollmann, and E. Snekkenes, Eds. Cham: Springer International Publishing, 2017, pp. 315–333, ISBN: 978-3-319-66402-6. [Online]. Available: https://doi.org/10.1007/978-3-319-66402-6_19.

[6] *Trustworthy Systems Supporting Server Signing. Part 2: Protection Profile for QSCD for Server Signing*, draft prEN 419 241-2:2017, version 0.15, Oct. 2017.

[7] *Smart-ID Technical Architecture*, version 6.1, 2017.

[8] *Smart-ID SecureZone Technical Architecture*, version 10.13, 2018.

[9] *Smart-ID Threshold Signature Engine Security Target*, version 2.4.0, 2018.

[10] *Administration Guide for SecureZone*, version 1.3, 2018.

[11] *Installation Guide for SecureZone*, version 1.3, 2018.

[12] *Smart-ID SecureZone monitoring guide*, version 1.1_v16, 2018.

[13] *Signer User Guidance information for SecureZone and TSE library operators*, version 1.0_v6, 2018.

[14] *Security Requirements for Cryptographic Modules*, FIPS PUB 140-2, NIST, May 2001. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-2.pdf.

[15] *Policy and security requirements for Trust Service Providers issuing certificates. Part 1: General requirements*, ETSI draft EN 319 411-1:2017, version 1.2.0, Aug. 2017. [Online]. Available: http://www.etsi.org/deliver/etsi_en/319400_319499/31941101/01.02.00_20/en_31941101v010200a.pdf.

[16] *Protection profiles for Secure Signature Creation Device — Part 2: Device with key generation*, prEN 14169-2:20125, Jan. 2012.

[17] *Policy and security requirements for Trust Service Providers issuing certificates. Part 2: Requirements for trust service providers issuing EU qualified certificates*, ETSI draft EN 319 411-2:2016, version 2.1.1, Feb. 2016. [Online]. Available: http://www.etsi.org/deliver/etsi_en/319400_319499/31941102/02.01.01_60/en_31941102v020101p.pdf.

[18] ETSI, *Electronic Signatures and Infrastructures (ESI): Cryptographic Suites*, 2014. [Online]. Available: http://www.etsi.org/deliver/etsi_ts/119300_119399/119312/01.01.01_60/ts_119312v010101p.pdf.

[19] S.-I. C. W. Group, *SOG-IS Crypto Evaluation Scheme Agreed Cryptographic Mechanisms*, May 2016. [Online]. Available: https://www.sogis.org/documents/cc/crypto/SOGIS-Agreed-Cryptographic-Mechanisms-1.0.pdf.

[20] *PKCS #1: RSA Cryptography Specifications Version 2.2*, RFC 8017 (Informational), IETF, Nov. 2016. [Online]. Available: https://tools.ietf.org/html/rfc8017.

[21] *Diffie-Hellman Key Agreement Method*, RFC 2631 (Informational), IETF, Jun. 1999. [Online]. Available: https://tools.ietf.org/html/rfc2631.

[22] *More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)*, RFC 3526 (Informational), IETF, May 2003. [Online]. Available: https://tools.ietf.org/html/rfc3526.

[23] *Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography*, FIPS SP 800-56A Rev. 2, NIST, May 2013. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar2.pdf.

[24] *Recommendation for Cryptographic Key Generation*, FIPS SP 800-133, NIST, Dec. 2012. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-133.pdf.

[25] *Specification for the Advanced Encryption Standard (AES)*, FIPS PUB 197, NIST, Nov. 2001. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf.

[26] *The Keyed-Hash Message Authentication Code (HMAC)*, FIPS PUB 198-1, NIST, Jul. 2008. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.198-1.pdf.

[27] *Secure Hash Standard (SHS)*, FIPS PUB 180-4, NIST, Aug. 2015. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf.

1.4 Terms and Abbreviations

Notation	Description
CA	Certificate Authority – see also Certificate Service Provider (CSP).
CC	Common Criteria
CGA	Certificate Generation Application – service which allows the Signer to obtain a Qualified Certificate for Electronic Signature, which binds together the Validation Data and the Person Identification Data of the Signer, from a Qualified Trust Service Provider.
CSP	Certificate Service Provider – service, which issues the certificates binding together the SVD and identity of Signer. See also Certificate Authority (CA).
DTBS	Data To Be Signed – the document, which the Signer wishes to sign. See also the asset D.DTBS.
DTBS/R	Data To Be Signed Representation – DTBS/R is generated from the Data To Be Signed (DTBS) with hash algorithm. See also the asset D.DTBS/R.
HSM	Hardware Security Module – trusted hardware component, which is providing the certified cryptographic functions.
HSM master key	Hardware Security Module master key – a root (or master) key is used to encrypt other keys that are in turn used to encrypt the actual data you want to protect. The master key can decrypt all of the other keys, and therefore (indirectly) all of the data.
ICT	Information and Communications Technology
J2EE	Java Platform Enterprise Edition – Java computing platform for development and deployment of enterprise software (network and web services).
JDK	Java SE Development Kit – software package, which includes the Java Virtual Machine (JVM) and the related libraries and utilities in order to run the Java applications.
JRE	Java Runtime Environment – the standard execution environment for the Java applications. See also JVM.
JVM	Java Virtual Machine – the standard execution environment for the Java applications. See also Java Runtime Environment (JRE).
keyUUID	Key Universally Unique IDentifier – D.Signing_Key_Id is referenced as keyUUID in some places since this is the name of the attribute in the developer documents and sources. This is the unique identifier of the signing keys and this is the id as well which the Signer is mapped to the keys by.
KTK	Key Transfer Key – the key which is used to encrypt cryptographic key material for transferring it from one Smart-ID system component to another component, over insecure communication channel. See also the asset D.KTK.

Notation	Description
KWK	Key Wrapping Key – the key which is used to encrypt cryptographic key material for the purposes of secure storage of the key material. See also the asset D.KWK.
OCS	Operator CardSet – The smart-cards used by the Hardware Security Module (HSM), which are used to authenticate operators.
PKI	Public Key Infrastructure
QSCD	Qualified Signature Creation Device – device, which produces the qualified signatures according to the reg. (EU) 910/2014 [4].
RA	Registration Authority.
SAP	Signature Activation Protocol – Cryptographic protocol for activating the signing keys in the Server-Signing solutions.
SCA	Signature Creation Application
SCD	Signature Creation Data – the private key used for creating electronic signatures. See also asset D.SCD.
SCD/SVD	Cryptographic key pair with the Signature Creation Data (SCD) as the private key and Signature Verification Data (SVD) as the public key.
Signer	The natural person, who is the owner of the key pair (SCD and SVD) and who is creating the digital signatures with the key pair.
SSCD	Secure Signature Creation Device
ST	Security Target
SVD	Signature Creation Data – the public key corresponding to the SCD for a signature, which can be used to verify the signature.
TEK	Transport Encryption Key – An AES-256 symmetric cryptographic key shared between the TOE and specific instance of TSE. It is used to protect the communication between TSE instance and SecureZone. TEK is created per key pair and has the same life cycle as key pair.
TOE	Target of Evaluation
TSE	Threshold Signature Engine – Smart-ID App TSE is the software component, which works within the Signer's environment and helps and assists Signer to follow the Threshold Signature Scheme Protocol (TSSP) and to use the Smart-ID SecureZone services for the key enrolment and signature creation.
TSF	TOE Security Functionality
TSFI	TOE Security Functionality Interface – The interface over which the TOE Security Functionality can be accessed and used or over which the data flows in either direction.
TSP	Trust Service Provider
TSSP	Threshold Signature Scheme Protocol – cryptographic protocol and algorithms followed by the Signer and TOE, in order to generate the distributed key pair of the Signer and later using the key pair to produce the signature of the Signer. The TSSP is defined in the peer-review published article [5].
UML	Uniform Modelling Language
VAD	Verification Authentication Data – signer's VAD is the data, which is input by the Signer in order to authenticate himself. Usually this is the PIN code of the Signer.
VM	Virtual Machine

1.5 ST Reference Identification

Title: Smart-ID SecureZone Security Target
Version: 2.7.0
Publication date: September 21, 2018

1.6 TOE Reference Identification

TOE identification/version: Smart-ID SecureZone version 10.3.5

1.7 Document changelog

Version	Date	Summary of changes
1.0.0	15.05.2017	First submission to the evaluation process
1.0.1	24.05.2017	1. Update to the CC version 3.1, release 5 2. Addition of the A.PRIVILEGED_USER 3. Improved definitions of the SFRs in the section 6.1
1.0.2	01.06.2017	1. Complemented Smart-ID system description in chapter 2 and fixed the product name spelling. 2. Added the description of the re-key functionality to the TOE with the corresponding SFPs in the section 6.1.2 and SF in the section 7.5.2. 3. Added the section 6.2 "Security Assurance Requirements" to the chapter 6. 4. Added the section 6.3 "SFR Dependencies Rationale" to the chapter 6. 5. Deleted the "Requirement Rationale" table from the chapter 6 because the table is no longer useful and the SFRs have the proper definitions in the section 6.1. 6. Clarified the user authentication and the roles in the TOE within the sections 6.1 and chapter 7.
1.0.3	15.06.2017	Fixed the problems outlined on the "Observation Report V1", observations 1 to 32. Detailed list of individual changes are listed in the response to the report.

Version	Date	Summary of changes
1.0.4	05.07.2017	1. Fixed the problems outlined on the "Observation Report V1", observations 33 to 51. Detailed list of individual changes are listed in the response to the report. 2. Fixed the typos and problems outlined by SK.
2.0.0	19.01.2018	Rewrite of the document to be more similar with the concepts of PP 419 241-2 [6].
2.1.0	02.03.2018	Fixed the problems outlined on the "Observation Report V3". Detailed list of individual changes are listed in the response to the report. Improved the document according to TÜViT's feedback from the meeting on February 8th 2018.
2.2.0	16.03.2018	Fixed the problems outlined on the "Observation Report V4". Detailed list of individual changes are listed in the response to the report.
2.3.0	21.05.2018	Fixed the problems outlined in the Observation Reports V5 and V6. Detailed list of individual changes are listed in the response to the reports.
2.4.0	22.06.2018	Fixed the problems outlined in the Observation Report V7. Detailed list of individual changes are listed in the response to the report. Updated the SecureZone reference version number to v10.3
2.5.0	30.07.2018	Fixed the problems outlined in the SZ ST Observation Report V8 and SZ AGD Observation Report V4. Detailed list of individual changes are listed in the response to the report. Removed the irrelevant reference to TSE in section 2.4.1.
2.6.0	19.09.2018	Fixes of the supported KWK, KTK and SHA-2 key sizes, according to ADV_IMP OR. Updated the database server version number in section 2.4.4 that was used during testing.
2.7.0	21.09.2018	Fixed the list of SFRs in section 8.1.1.2 SF.AccessControl, under point 2. Fixed the version number of the TOE as 10.3.5.

2 System Overview

This chapter provides an informal overview of the digital signatures, Smart-ID Threshold Signature Scheme Protocol and the Smart-ID SecureZone as the TOE of this ST document. The formal Security Problem Definition using the CC terms, is given in the next chapters. However, where appropriate, references are made to the definitions in the following sections of the document.

2.1 Introduction to the Smart-ID system

The invention of the digital signatures and the Public Key Infrastructure (PKI) has enabled society to use convenient authentication and signature features. For example, when digital signature technology is combined with the smart-cards, the secure storage of the private keys can be implemented. Together with the PKI technology, the Secure Signature Creation Devices (SSCDs) have been developed by the Information and Communications Technology (ICT) industry. With such solution, the protection of the Signer's private key is handled by the Signer itself. However, as the features of the personal computing devices have been evolved, the usage of such special purpose devices has become more and more inconvenient. The ICT industry has been searching for alternative solutions. One of such solutions is the server-signing services, where the protection of the private key of the Signer is entrusted to the server-signing service provider.

The Smart-ID system has been developed to provide alternative solution for the digital signature creation, where the risk and responsibility to secure the private key is no longer placed to the single system participant, but it is shared between multiple system components. With the application of the cryptographic threshold signature protocols, the private key can be generated in shares. In order to use the private key, to create the digital signatures, the shares don't need to be combined in the single physical location. Instead, the individual shares are used to create the shares of the signature. Only when all shares of the signature are combined, the compound signature is achieved. With such kind of protocol, overall risks and technical threats can be greatly reduced.

The current document describes the Smart-ID TSSP and the Smart-ID SecureZone, which is the server-side implementation of this protocol and the Target of Evaluation (TOE) of this ST document.

2.2 Overview of the TOE

This section describes the TOE and explains its intended usage.

The TOE described in this Security Target (ST) is inspired by but is not strictly conformant to the PP 419 241-2 [6]. The reason for non-strict conformance is due to differences in the underlying technical solutions of the TOE and classical server signature solutions. The

differences come from the TOEs usage of the Smart-ID TSSP ([5]). Otherwise, the same terminology and methodology as in the protection profile is used in the current ST document for describing the TOE. There are some informative references to the comparable assets and threats to the PP 419 241-2 [6] for the purpose of quicker grasp of the ST document and straightforward comparison.

2.2.1 TOE definition

The TOE is the computer software product "Smart-ID SecureZone". It is a Java application server package, which implements the server-side functions of the TSSP for the Signer and the management functions for the administrators. The Signer, who follows the client-side functions of the TSSP, can use the TOE services to enroll new key pairs, create digital signatures and to destroy the key pairs.

The important distinction here is that the TOE alone doesn't create the whole digital signature on behalf of the Signer, but they both participate in the cryptographic protocol. The TSSP is further explained in the section 2.3.

2.2.2 TOE type

The TOE is a software component, which implements the server-side functions of the Threshold Signature Scheme Protocol TSSP to activate a signature. It is deployed in a dedicated tamper protected environment, that is connected to the Hardware Security Module (HSM) via a trusted channel. It uses the Signature Activation Data (SAD) from the signer to complete the signature computation with the HSM.

Together, the mobile client, the TOE and the HSM are a Qualified Signature Creation Device (QSCD).

2.2.3 TOE usage and major security features

The TOE is intended to be used as a component of a QSCD system to conduct the following functions:

1. Creation of Qualified Electronic Signatures, complying with eIDAS regulation reg. (EU) 910/2014 [4];
2. Enrolment and destruction of the Signer's key pair;
3. Security management and access control functions.

The high-level security features of the TOE are similar to the high-level security features of traditional QSCDs. The features are grouped into the following subsections, categorized according to the abovementioned main usage functions of the TOE.

2.2.3.1 Enrolment of the components of the new key pair of the Signer and other keys

1. Import of the server's part of the private key of the Signer. This is the asset D.serverPart.
2. Usage of the HSM to generate the server's share of the private key of the Signer. This is the asset D.serverShare.
3. Generation of the compound public key of the Signer. This is the asset D.SVD.
4. Generation of D.KTK RSA key (by HSM) for encrypting transferred keys between TSE and TOE.
5. Generation of D.TEK (by TOE) to protect the communication between the TSE and TOE.
6. Generation of D.KWK AES key (by HSM) for wrapping key material on the TOE database.

7. Generation of D.DEK AES key (by TOE) to encrypt certain database fields.

More details can be found in the section 2.5.3.2 Handling of cryptographic material and algorithms

2.2.3.2 Signature creation

1. Creation of the server's part of the signature of the Signer. This is the asset D.server SignaturePart.
2. Creation and validation of the applicationSignatureShare of the Signer from the D.applicationSignaturePart and the D.serverSignaturePart. This is the asset D.applicationSignatureShare.
3. Usage of the HSM to create of the server's share of the signature of the Signer. This is the asset D.serverSignatureShare.
4. Creation and validation of the compound signature of the Signer. This is the asset D.signature.

2.2.3.3 Destroying the components of the key pair of the Signer

1. Destroying the shares of the private key of the Signer, the assets D.serverPart and D.serverShare.

2.2.3.4 Security management functions

TOE also has the following management features:

1. Starting the TOE instance and securely connecting to the HSM to load the encryption keys for the TOE database.
2. Generation of D.KWK, D.KTK and D.DEK encryption keys.
3. Batch pre-generation of the multiple D.serverShares for performance reasons.
4. Re-key process initiated by the CA that enables generating new key-pair and the corresponding certificate for an existing Signer (see also table 13). The TOE is involved in the process by:
   4.1 re-generating and rewriting the D.serverShare of the Signer's private key;
   4.2 re-generating and returning the new Signer's compound public key D.SVD.

2.2.3.5 Authentication, Access Control and Security audit generation functions

TOE also has the following security features:

1. Authentication – This function provides different methods to authenticate users and protect the assets of the TOE.
2. Access control – Different users have access to their different assets and allowed operations.
3. Security audit generation – The audit records of the important system events are generated by standard Java toolset and the audit is exported to external system.

More details about them can be found in the section 2.5.3.1 TOE management and access control

2.2.3.6 Protecting communication with external components

TOE use trusted path (encryption) to communicate with the Smart-ID App TSE. On the other hand the TOE communicate with HSM and Database with vendor-specific secure channel.

More details about them can be found in the section 2.5.3.3 Protecting communication with external components

2.2.3.7 Functions not present in TOE

The TOE only provides the key pair related security functions and it doesn't have any features related to the identity proofing, Signer registration, certification issuing and other features, which are commonly required by the full-scale PKI system. So, in order to establish the larger PKI system, TOE will interface and work with the following external trusted IT systems:

1. Registration Authority (RA) is responsible for identity proofing of the Signer. The RA will use either existing digital identities of the Signer or performs the identity proofing procedures to verify the government issued identity document in person. The RA will then forward this information to the CA, so that CA can issue the certificate binding together the identity and the D.SVD of the Signer.
2. CA is responsible for issuing qualified certificates to the Signer. CA will receive the identity information from the RA and the D.SVD from the TOE.

TOE places certain requirements to the security level of such functions to be provided by external trusted IT systems, for example, TOE requires that the CA issues qualified certificates.

2.3 Threshold Signature Scheme Protocol (TSSP)

2.3.1 Introduction

The TSSP is the protocol to be followed by the Signer and the TOE, in order to generate the key pair of the Signer (the assets D.SCD and D.SVD), which is usable only when both Signer and the TOE are participating in the protocol. The private key of the key pair of the Signer (the asset D.SCD) is generated in shares. It is done in such a way, that multiple shares of the the private key (the assets D.clientPart, D.serverPart, D.serverShare) are separately generated and they are independently protected by Signer (the asset D.clientPart) and the TOE (the assets D.serverPart and D.serverShare).

In order to actually create the digital signature of the Signer, those individual shares of the private key have to be used by their respective holders to create the shares of the signature (the corresponding assets D.applicationSignaturePart, D.serverSignaturePart, D.serverSignature Share). Those shares of the signature must then be combined and the resulting compound signature (the asset D.signature) is then verifiable with the public key of the Signer (the asset D.SVD).

The TSSP is fulfilling the same kind of purposes as the Signature Activation Protocol (SAP) from the PP 419 241-2 [6] and provides the same security capabilities and in some way, TSSP can be seen as the instance of the SAP. However, the TSSP also includes the key pair enrolment protocol and provides additional unique security capabilities to Signer and TOE. Therefore, we refer to the TSSP in this ST document.

The next sections give the high-level abstract overview, how the TSSP works between the Signer and the TOE. Note that some technical details are omitted and simplified from these sections, in order to keep the description as short as possible. Please refer to the peer-reviewed paper [5] in order to get all the mathematical and cryptographical details along with

the security proofs. Also, please refer to the architecture documents [7] and [8] in order to get all the implementation details of the TOE.

2.3.2 Key pair generation process

The high-level process for the key pair generation of the Signer is shown in the Figure 1 with the UML sequence diagram. In the following sections, the components and messages shown in the diagram are explained.

2.3.2.1 Actors and components

- Signer – This is the natural person, who is using the Smart-ID App Threshold Signature Engine (TSE) and the TOE services to generate, protect and use the key pair, which is split into multiple shares according to the TSSP. Signer keeps the knowledge-based secret asset D.PIN.
- Smart-ID App TSE – This is the software component, which is running on the personal mobile device of the Signer (phone, tablet or other smart-device). The mobile device is under the Signer's control and is helping Signer to generate the app's share of the key pair and to protect it. The Smart-ID App TSE implements the client-side functions of the TSSP. The security functions of the Smart-ID App TSE is evaluated according to the separate ST document [9].
- Smart-ID SecureZone (TOE) – This is the software component, which is the TOE of the current ST document. The TOE implements the server-side functions of the TSSP. The TOE allows Signer to generate, protect and use the key pair, which is split into multiple shares according to the TSSP.
- Smart-ID SecureZone database – This is the database, which is used by TOE to store user data and TSF data. Sensitive security attributes are stored with HSM proprietary encryption or with TOE implemented encryption.
- Smart-ID SecureZone HSM – This is the trusted hardware component, which is providing the certified cryptographic functions to the TOE, such as key share generation and creation of the signature share.

2.3.2.2 Process steps

TSSP key pair generation steps (the numbers correspond to the messages on the sequence diagram):

1. SZ operator asks SZ to pre-generate the server's shares, so that registration of new Signers is quicker.
2. SZ asks HSM to generate the new server's share of the key pair (D.serverShare).
3. HSM generates the new server's share of the key pair (D.serverShare and D.serverModulus).
4. SZ receives the encrypted blob of the private key (D.serverShare) and the public key of the key pair (D.serverModulus).
5. SZ stores the private key (D.serverShare) and the public key of the key pair (D.serverModulus) in the SZ database and marks them free to be used. The private key is stored and transferred encrypted.
6. Signer asks the Smart-ID App TSE to start generating the new key pair.

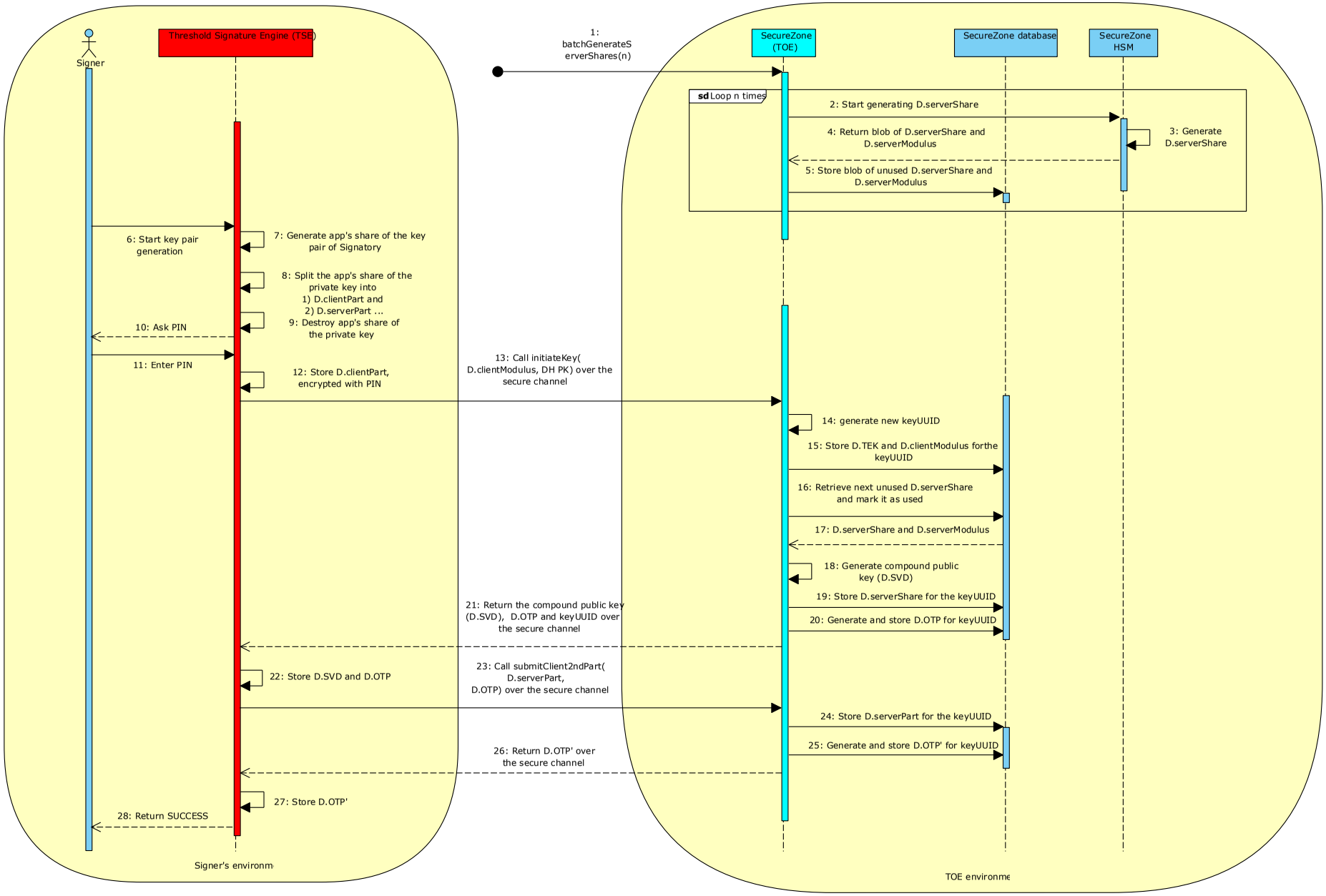

Figure 1. Overview of the enrollment procedure in the TSSP.

7. TSE generates the app's share of the key pair. The key pair consists of the private key and the public key (the asset D.clientModulus). TSE generates an ephemeral Diffie-Hellman key pair for D.TEK establishment.
8. TSE mathematically splits the private key of the generated key pair into two parts, using additive sharing method. The individual parts cannot be used to deduce information about the whole private key.
   8.1 D.clientPart is the part, which is stored and protected within the TSE
   8.2 D.serverPart is the part, which is to be transmitted to the SZ
9. TSE securely destroys the private key of the generated key pair.
10. TSE asks Signer to enter the D.PIN to derive the encryption key, which is used to encrypt the locally stored D.clientPart. The D.PIN is the knowledge-based factor, which is used to secure the TSSP.
11. Signer enters the D.PIN.
12. TSE uses the D.PIN to derive the encryption key and to encrypt the D.clientPart. The encryption is done in a way that no validation information about the cryptogram is stored. The D.PIN itself is not stored within the Smart-ID App TSE.
13. TSE initiates the initiateKey() operation (see also table 13) in the SZ, transmitting the D.clientModulus and the Diffie-Hellman public key (for establishing the D.TEK) to the SZ.
14. SZ receives the D.clientModulus and the client's Diffie-Hellman public key. SZ assigns fresh unique D.Signing_Key_Id to the new key pair of the Signer, executes the server-side steps of Diffie-Hellmann key exchange and generates D.TEK.
15. SZ stores D.clientModulus and D.TEK in the database.
16. SZ marks the next unused D.serverShare and D.serverModulus as used and retrieves them from database.
17. SZ receives the D.serverShare and D.serverModulus from database.
18. SZ generates the compound public key (D.SVD) by mod-multiplying together D.clientModulus and D.serverModulus
19. SZ stores the D.SVD in the database.
20. SZ generates the one time password (D.OTP) and stores it in the database.
21. SZ returns the D.SVD, D.OTP, D.Signing_Key_Id and Diffie-Hellmann key exchange material over the secure channel to the Smart-ID App TSE. The channel is secured by encrypting the data with the newly generated D.TEK.
22. TSE decrypts the response by using the D.TEK, verifies the Diffie-Hellmann key exchange material and stores the D.SVD and D.OTP.
23. TSE initiates the submitClient2ndPart() operation (see also table 13) in the SZ, by transmitting the D.serverPart and D.OTP over the secure channel to the SZ.
24. SZ stores the D.serverPart in the database.
25. SZ generates the new value of one time password (D.OTP) (OTP') and stores it in the database.
26. SZ returns the new value of one time password (D.OTP) (OTP') over the secure channel to the Smart-ID App TSE.
27. TSE decrypts the response by using the D.TEK and stores the new value of one time password D.OTP.
28. TSE shows the Signer the success message about the new key pair generation.

2.3.3 Signature generation process

The high-level signature creation process is shown in the Figure 2 with the UML sequence diagram. The process involves additional component, Signature Creation Application (Signature Creation Application (SCA)). The SCA is the general purpose trusted software application, which is used by the Signer in order to prepare and to create the digitally signed documents. Such features are not included in the Smart-ID App TSE or the TOE itself, in a similar way as the function for creation of digital documents are not included in the QSCDs.

2.3.3.1 Actors and components

- Signer – This is the natural person, who is using the SCA, Smart-ID App TSE and the TOE services to digitally sign the document.
- Signature Creation Application – This is the general purpose trusted software application, which is handling the technical issues with creating the well-formatted and -encoded digital documents, computing the DTBS/R and requesting the digital signature of the DTBS/R from the Signer.
- Smart-ID App TSE – This is the trusted software component, which is installed on the personal mobile device of the Signer (phone, tablet or other smart-device). The mobile device is under the Signer's control and is helping Signer to use the app's share of the key pair and to create the application's part of the signature. The Smart-ID App TSE implements the client-side functions of the TSSP. The security functions of the Smart-ID App TSE is evaluated according to the separate ST document [9].
- Smart-ID SecureZone (TOE) – This is the software component, which is the TOE of the current Security Target document. The TOE implements the server-side functions of the TSSP. The TOE allows Signer to use the shares of the key pair, which are stored in the TOE and to create the server's part of the signature and the compound signature.
- Smart-ID SecureZone database – This is the database, which is used by TOE to store user data and TSF data. Sensitive security attributes are stored with HSM proprietary encryption or with TOE implemented encryption.
- Smart-ID SecureZone HSM – This is the hardware component, which is providing the certified cryptographic functions to the TOE, such as key share generation and signature share generation.

2.3.3.2 Process steps

TSSP signature creation steps (the numbers correspond to the messages on the sequence diagram):

1. Signer asks the SCA to digitally sign the document.
2. SCA formats and encodes the document and computes the D.DTBS/R, which corresponds to the data to be signed.
3. SCA computes the verification code (D.VC) of the D.DTBS/R. The verification code is the short representation of the whole digest D.DTBS/R and allows Signer to verify, if he is agreeing with the correct signature request on the Smart-ID App TSE.
4. SCA displays the D.VC to the Signer and asks to verify it, when displayed by the Smart-ID App.
5. At the same time, SCA requests the signature of the D.DTBS/R from the Smart-ID App TSE.

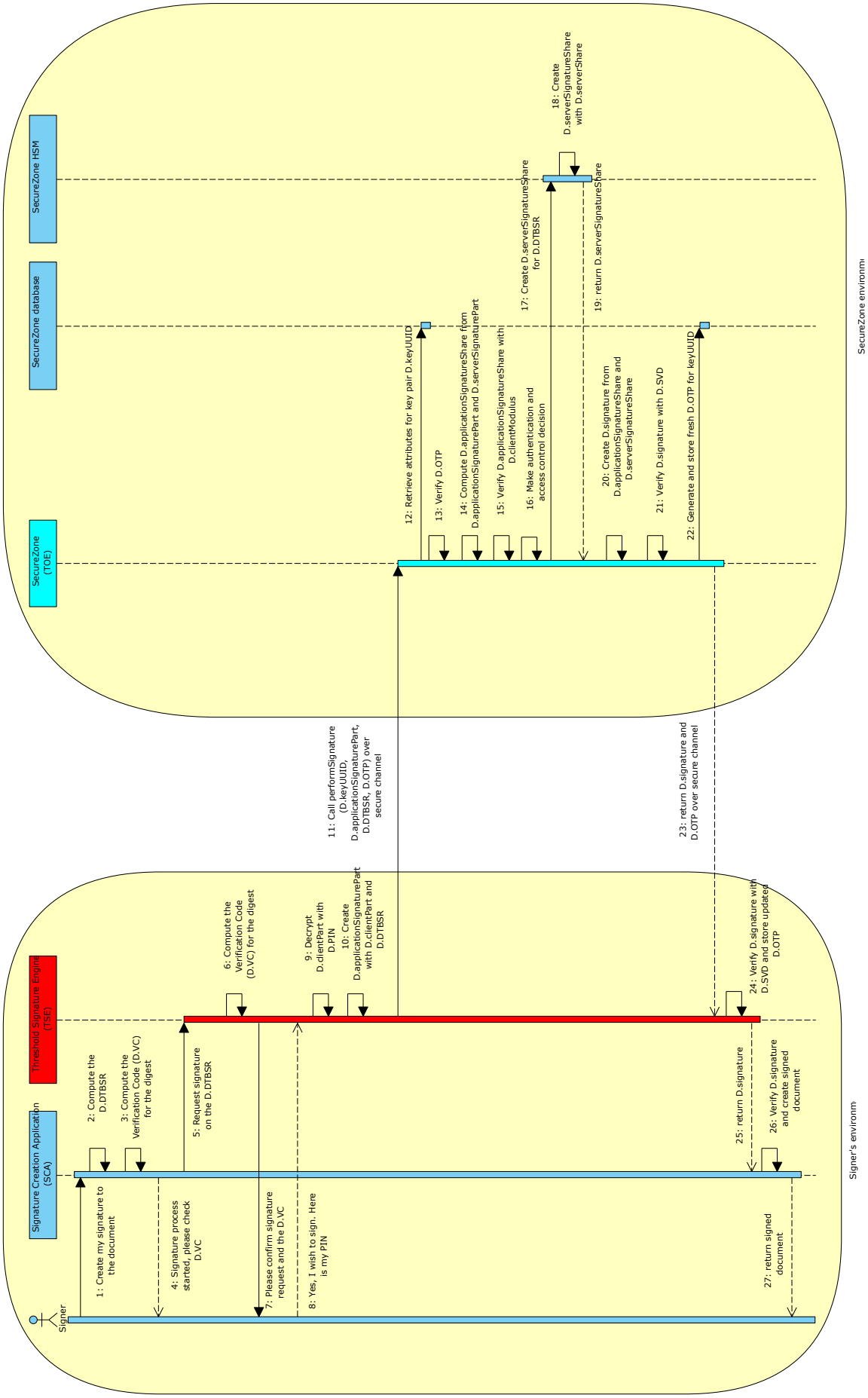

Figure 2. Overview of the signing procedure in the TSSP.

6. TSE receives the request and computes the verification code (D.VC) along with the D.DTBS/R.
7. TSE informs the Signer of the new signing request and displays the D.VC and asks for the Signer's confirmation and the D.PIN.
8. Signer verifies that the TSE displays the same D.VC as the SCA and agrees with the request. Signer enters the D.PIN to the TSE.
9. TSE uses the D.PIN to decrypt the D.clientPart. Note that TSE does not verify if the entered D.PIN is correct or if the decrypted D.clientPart is valid or correct. There is no way for the TSE to validate the entered D.PIN locally, without contacting the TOE.
10. TSE uses the decrypted D.clientPart to create the signature share D.applicationSignature Part with the D.DTBS/R.
11. TSE initiates the performSignature() operation (see also table 13) in the TOE over the secure channel (by encrypting the data with D.TEK key), along with the following data: D.Signing_Key_Id, D.applicationSignaturePart, D.DTBS/R, D.OTP.
12. TOE retrieves attributes for the keypair D.Signing_Key_Id from the database.
13. TOE verifies that (D.OTP) for that particular D.Signing_Key_Id is valid. This gives us the possession-based authentication factor.
14. TOE uses the D.serverPart to create the signature share D.serverSignaturePart with the D.DTBS/R and then uses the signature parts D.applicationSignaturePart and D.server SignaturePart to create the signature share D.applicationSignatureShare.
15. TOE verifies if the signature share D.applicationSignatureShare is valid, with the D.clientModulus.
16. TOE makes the authentication and access control decision. If the signature share is not valid, the signature completion request is cancelled. This gives use the knowledge-based authentication factor since the Signer had to use the correct D.PIN to decrypt the local D.clientPart.
17. TOE sends the D.DTBS/R to the HSM and asks for the creation of signature share with the D.serverShare.
18. HSM creates the signature share D.serverSignatureShare.
19. TOE receives the D.serverSignatureShare and verifies it with D.serverModulus and D.DTBS/R.
20. TOE creates the compound signature D.signature from the signature shares D.application SignatureShare and D.serverSignatureShare.
21. TOE verifies, if the compound signature D.signature is valid with the D.DTBS/R and D.SVD.
22. TOE generates fresh (D.OTP) and stores it in database.
23. TOE returns the compound signature D.signature and updated D.OTP to the TSE over the secure channel (by encrypting the data with the specific instance of D.TEK).
24. TSE decrypts the response by using the D.TEK key and receives D.signature and D.OTP. TSE verifies, if the D.signature is valid with the D.DTBS/R and D.SVD.
25. TSE returns the compound signature D.signature to the SCA and displays a notification to the Signer that the signature has been created successfully.
26. SCA receives the compound signature D.signature, verifies, if it is valid with the D.DTBS/R and D.SVD and creates the digitally signed document with the Signer's signature.

27. SCA returns the digitally signed document to the Signer.

2.4 Required non-TOE hardware/software/firmware

This section lists the hardware and software components, which are required in order to successfully and securely run the TOE. In the previous sections, we have explained the major security functions of the TOE and described how the TSSP works. Note that the external components described on the sequence diagrams in the sections 2.3.2 and 2.3.3, such as the SCA are not strictly required to operate the TOE and therefore, they are not listed in this section.

2.4.1 Smart-ID App Threshold Signature Engine

The TOE must be used in conjunction with the Smart-ID Threshold Signature Engine TSE software library, or another application fulfilling the requirements set in the ST document [9] and evaluated to correspond with the EAL2 level, according to Common Criteria Part 3 [3].

2.4.2 Server hardware and operating system

TOE is independent of hardware and operating system, the TOE security functions do not depend on the security functions of the underlying operating system.

TOE has been tested with 64 bit Centos 7 Linux and is compatible with any 64-bit Linux distribution released after June 2015, containing Linux kernel 3.10 or later.

The server hardware must be capable enough to run the operating system and the JVM along with the TOE software image. Most commonly, a generic x86-based server with 32-bit or 64-bit, 1.0 GHz or faster CPU and 4 GB or more RAM is used.

2.4.3 J2EE application server

TOE is independent of the application server and execution environment where it is used, as long as the execution environment is a Java Virtual Machine (VM) compliant with the Java Platform Enterprise Edition (J2EE) requirements. Though the TOE security functions do not depend on the security functions of the application server or execution environment, it should be ensured that the TOE is the only application, which is running on the JVM.

TOE has been tested with OpenJDK 8 and is compatible with any up-to-date version of JRE declaring compatibility with Java SE 8.

2.4.4 Database

TOE is using general purpose database to store the operational data. The sensitive fields in the database are encrypted and they are protected with the integrity protection mechanisms. Therefore, the security features of the database are not relevant for the security of the TOE.

TOE has been tested with PostgreSQL 10.4 and is compatible with PostgreSQL versions 9.6 and later.

2.4.5 Hardware Security Module

The HSM supplies its own set of security functions and has to be certified to be compliant with the QSCD requirements according to eIDAS reg. (EU) 910/2014 [4]. The HSM is regarded as the trusted device and TOE is relying on the security functions of the HSM in order to fulfil subset of the security objectives of the TOE.

TOE has been tested with Thales nShield Connect 6000+ HSM, with part number NH2068 and with the following version information:

- Hardserver version 2.92.1
- Client libraries: Generic stub version 3.30.5, NFKM and RQCard version 1.86.1, and PKCS#11 version 2.14.1
- Client utilities version 2.54.1

TOE is compatible with Thales nShield Connect series HSMs that are also certified to be compliant with the QSCD requirements according to eIDAS.

2.5 Description of the TOE

2.5.1 Physical scope of the TOE

The following physical items make up the physical scope of the TOE:

1. Smart-ID SecureZone service software package, delivered within a war-archive file
2. Smart-ID SecureZone administrative command line interface (CLI) tool, delivered within a war-archive file
3. Installation and Administration Guides for SecureZone, consisting of:
   - 3.1 Administration Guide for SecureZone [10], delivered in pdf format
   - 3.2 Installation Guide for SecureZone [11], delivered in pdf format
   - 3.3 Smart-ID SecureZone monitoring guide [12], delivered in pdf format
   - 3.4 Signer User Guidance information for SecureZone and TSE library operators [13], delivered in pdf format
   - 3.5 Smart-ID SecureZone Technical Architecture [8], delivered in pdf format

Each part of the TOE physical scope is delivered via a secure file transfer system. The secure delivery procedure of the items constituting the physical scope of the TOE must include verification of the checksums of all the delivered components and verification of the correspondence of version numbers in the TOE documentation and the .war files.

2.5.2 Components outside of the physical scope of the TOE

The TOE is physically represented by the Smart-ID SecureZone software, written in Java and packaged into a war-archive. The war-archive is installed and executed in the J2EE application server. Because the TOE does not rely on the security features of the J2EE VM or the server operating system, those components are outside of the physical scope of the TOE.

The TOE exposes the API to the outside world, which can be used by external users to initiate communication to TOE. This API is TOE Security Functionality Interface (TSFI). TOE also uses the HSM and database APIs and because the information retrieved over those interfaces also influence TOE security functionality, they are considered TSFIs as well.

The TOE uses the HSM for the cryptographic operations. The HSM is required to be trusted and the security functions of the HSM are required to be certified to be compliant with the QSCD requirements according to eIDAS reg. (EU) 910/2014 [4]. Because HSM is already certified, it is not included in the scope of the TOE and the security functions of the HSM are not evaluated according to the current ST.

The TOE uses the external database to store the cryptographic key material and to keep track of the key usage information. The TOE encrypts the sensitive data fields in the database and utilises the integration protection techniques, so that the external database component can be un-trusted and cannot influence the TOE Security Functionality (TSF).

The overview of the physical scope of the TOE is given in the Figure 3.

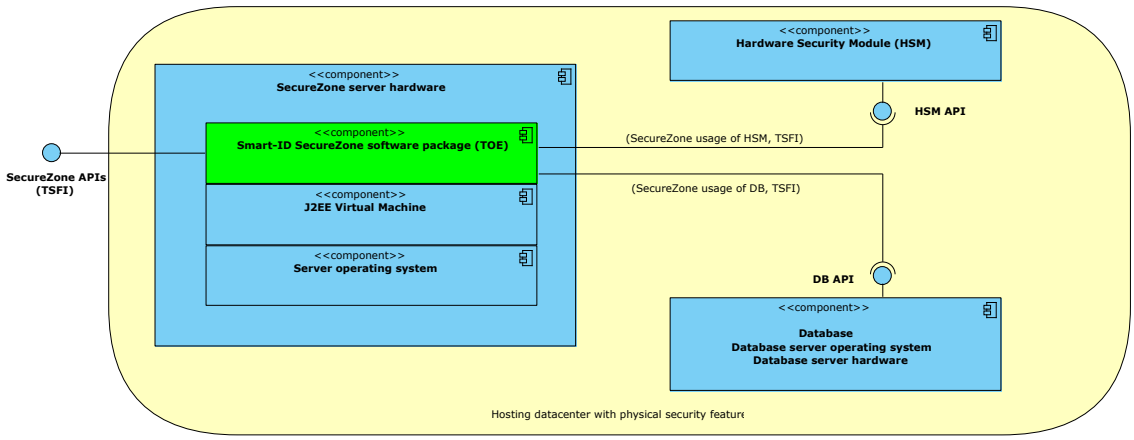

Figure 3. TOE physical scope.

2.5.3 Logical scope of the TOE

This section describes the logical scope of the TOE.

2.5.3.1 TOE management and access control

1. Authentication – This function provides different methods to authenticate users and protect the assets of the TOE. Technical functions of the TOE, which do not require personalised user identification/authentication and strict access control are not authenticated (cf. 7.2.5 or the monitoring interface).

   Other operations claim authentication of the users. The S.App and S.Admin authenticate with posession-based data (user-name and password). The S.Signer uses two-factor authentication based on knowledge and posession secret asset (D.PIN and D.OTP). The authentication process based on the TSSP which is described in the section 2.3. There is a lock mechanism to restrict the unsuccesful signer authentication by locking their key pair for a defined time.

2. Access control – Different users have access to their different assets and allowed operations. Anonymous users are allowed to perform some operations, which do not require authentication and authorisation (for example, querying the status of the key pair or not sensitive information).

   The access of Signer depends on his authentication method. In case of authentication with possession-based authentication factors, when the Smart-ID App is performing technical operations on behalf of the Signer and the App doesn't request authorisation with the entry of the D.PIN from the Signer, TOE only allows to perform technical operations (creating a signature is not possible).

   The key pair owners (Signers) are allowed to perform the key pair operations on their own key pair. In case the Signer is authenticated with possession-based and the knowledge-based authentication data, the TOE allows to complete the signature.

Privileged users can perform key pair operations on any key pair, however, the list of operations is limited to only specific methods. Privileged users are not allowed to invoke signature completion at all.

All the rules are described in more detail within the section 7.2 Security Function Policies (SFP). On the other hand the section 8.1.1.2 SF.AccessControl desribes the details of the TOE Access Control mechanism.

3. Security audit generation – The audit records of the important system events are generated by standard Java toolset and the audit is exported to external system.

2.5.3.2 Handling of cryptographic material and algorithms

1. Key generation – TOE uses the FIPS 140-2 [14]-certified HSM to perform most of the key generation operations. In case the HSM doesn't support generation and management of particular key type, TOE is generating that by himself. The following keys are generated:

   - D.SVD (by TOE implementing the TSSP [5]) using modulus multiplication of D.clientModulus and D.serverModulus,
   - D.serverShare RSA key (by HSM),
   - D.KTK RSA key (by HSM),
   - D.TEK (by TOE),
   - D.KWK AES key (by HSM) and
   - D.DEK AES key (by TOE).

   Further details can be found in 8.1.2.1 SF.KeyGen – Key generation section.

2. Re-key process – TOE allows the CA to use the reKey operation during which the new D.serverShare RSA key is generated (by HSM) and the new corresponding D.SVD is created (by TOE) and associated with an existing D.Signing_Key_Id.

   See also section 7.2.1 Operations, table 13, further details about the generation of the mentioned keys can be found in 8.1.2.1 SF.KeyGen – Key generation section.

3. Batch pre-generation of the multiple D.serverShare – For performance reasons the TOE allows Administrator to use the batchGenerateServerShares method to pre-generate the new batch of D.serverShare assets not associated with any existing D.Signing_Key_Id. It will be used during the new key-pair enrolments so that Signer enrolment can be done quicker.

   See more in section 7.2.1 Operations, table 14.

4. Storing and protection of keys – The following cryptographic keys are stored in TOE database, protected by HSM master key: D.KTK, D.KWK and D.serverShare.

5. Cryptographic algorithms and operations – The following cryptographic algorithms are used in the TOE processes: computation of the signatures implementing the TSSP, creation and verification of RSA signatures and encryption/decryption of JWE messages for transmission and database storage.

6. Key destruction – The TOE destroys the following cryptographic keys after they are no longer used: D.serverPart, D.serverShare, D.DEK, D.TEK, D.KWK, D.KTK.

The section 8.1.2 Handling of cryptographic material and algorithms describes the details of the mechanisms of cryptographic material and algorithms.

2.5.3.3 Protecting communication with external components

1. Trusted path with the user – TOE uses JWE messages for communicating with the Smart-ID App TSE. JWE messages are encrypted with the D.TEK and they are integrity protected.
2. Secure channel with external components – TOE uses vendor-specific proprietary communication channel when connecting with HSM or database, such as nCipher impath and PostgreSQL connections. Those methods provide the cryptographic checksum validation of the integrity for the transmitted data. When TOE detects the modifications and integrity errors with the transmitted data, it aborts the operation.

2.5.4 Features outside of the logical scope of the TOE

The TOE only provides the key pair related security functions and it doesn't have any features related to the identity proofing, Signer registration, certification issuing and other features, which is commonly required by the full-scale PKI system.

Other features, which may be installed and configured on the SecureZone server hardware as well, are not included in the logical scope of the TOE. For example, the following features are not included in the logical scope of the TOE.

1. Software included in the operating system libraries and the applications, which are required to run and manage the SecureZone server.
2. HSM software packages, in case of the Thales nShield HSM, the Thales nShield nCore API libraries, the Thales nShield hardserver software and the file store for the Thales nShield Security World.
3. Database software packages and libraries, which are required to connect to the external database server.

3 Conformance Claims (ASE_CCL)

3.1 CC Conformance

As defined by the references [1], [2] and [3], this TOE conforms to the requirements of Common Criteria version 3.1, revision 5.
Particularly: This Security Target claims to be Common Criteria Part 2 [2] and Common Criteria Part 3 [3] conformant.

3.2 Package conformance

This ST conforms to assurance package EAL4 augmented by AVA_VAN.5 defined in [3].

3.3 PP Conformance

This ST does not claim conformance to any PP.

3.4 EU regulation conformance

This ST claims conformance to reg. (EU) 910/2014 [4] with fulfilling the following organisational policy requirements defined in section 4.5:

1. P.SCD_Confidential
2. P.SCD_Unique
3. P.Sig_unForgeable
4. P.SCD_userOnly
5. P.DTBS_Integrity
6. P.TSP_Qualified
7. P.SCD_Backup
8. P.DTBS/R_Unique
9. P.TSP_QCert

4 Security Problem Definition (ASE_SPD)

This section gives the list and definitions of the conceptual data assets, which are used to describe the threats and security objectives of the TOE. Not all of the data assets are managed or protected by the TOE itself. For more details, please refer to the list of user attributes and security attributes in the section 7.1.

4.1 Assets

Name	Description	Security
D.application SignaturePart	Share of the signature of D.DTBS/R, which is computed by the Signer with the D.clientPart. It is not possible to validate the D.applicationSignature Part with any public key. This is one part of the D.Reference_Signer_Authentication_Data since when combined with D.serverSignaturePart, it will be the proof that the Signer used a correct PIN on the client side.	confidentiality, integrity
D.application Signature Share	Share of the signature of D.DTBS/R, which is created with the private key corresponding to the compound of D.clientPart and D.serverPart. That corresponding private key does not exist, therefore this signature share is instead created from the signature shares D.applicationSignaturePart and D.serverSignaturePart. The D.applicationSignatureShare can be validated with D.clientModulus.	confidentiality, integrity
D.Audit_Data	Audit records generated by the TOE and stored and protected outside of the TOE.	confidentiality, integrity, authenticity
D.Authorisation_ Data	It is the data used by the TOE to authorise the signature computation. The D.Authorisation_Data is part of the D.SAD. The TOE verifies the D.SAD before the signature computation. D.Authorisation_Data consists of D.Signing_Key_Id, D.DTBS/R and D.application SignaturePart.	confidentiality, integrity

Name	Description	Security
D.clientModulus	Data, which can certify the integrity of D.application SignatureShare. This is the public part of the D.clientShare/D.clientModulus key pair.	integrity
D.clientPart	Part of the D.SCD. It is generated and protected by Signer's PIN in the Smart-ID App sandbox in the Signer's mobile device. This also serves as one of the possession-based authentication factors used to authenticate the signer.	confidentiality, integrity
D.clientShare	Part of the D.SCD. It is generated in the Smart-ID App sandbox in the Signer's mobile device and then mathematically divided into D.clientPart and D.serverPart and then deleted.	confidentiality, integrity
D.DEK	Symmetric cryptographic key, which is used by TOE to encrypt and to integrity protect some database fields. D.DEK is generated and used by TOE itself, and it is wrapped with the D.KWK.	confidentiality, integrity
D.DTBS	A Data which the Signer intends to sign in the SCA.	integrity
D.DTBS/R	A representation of a set of data, which the Signer intends to sign. This is the digest value, generated with the hash algorithm from the D.DTBS.	integrity
D.KTK	Asymmetric encryption/decryption key pair, which is used to wrap the key material during the transmission from TSE to TOE. TOE uses the HSM to generate and protect the key.	confidentiality, integrity
D.KWK	Symmetric encryption/decryption and integrity protection key, which is used to wrap the key material in the TOE database. TOE uses the HSM to generate and protect the key	confidentiality, integrity
D.OTP	One time password. Password token, which is updated and given to the TSE by the SecureZone for each subsequent key pair operation.	confidentiality, integrity
D.PIN	PIN is known by Signer and is entered to the TSE by Signer to authorise each signing operation. The D.PIN itself is never stored within TSE or TOE and never transmitted. Instead, the D.PIN is only used to derive the encryption/decryption key, which is used to protect the D.clientPart, when stored in the Signer's mobile device.	confidentiality

Name	Description	Security
D.Privileged_ User	A set of data, that uniquely identifies a Privileged User within the TOE. In SecureZone there are two types of privileged users: 1. Administrator 2. CA	confidentiality, integrity
D.Random	Source of the random numbers, which are used to generate the encryption keys.	confidentiality, integrity
D.Reference_ App_ Authentication_ Data	This is the subset of D.Reference_Signer_Authentication_Data. This data is used by the TOE to authenticate the Signer's mobile device where the Smart-ID App TSE has been installed, i.e. this is the data related with the Signer's possession-based authentication factor. It consists of: 1. D.OTP 2. D.Signing_Key_Id	confidentiality, integrity
D.Reference_ Privileged_ User_ Authentication_ Data	A set of data used by TOE to authenticate the privileged user.	confidentiality, integrity
D.Reference_ Signer_ Authentication_ Data	This is the set of data used by TOE to authenticate the signer. It contains all the data and keys used by the TOE to authenticate the signer. This consists of the following data: 1. D.applicationSignaturePart 2. D.DTBS/R 3. D.Reference_App_Authentication_Data	confidentiality, integrity
D.SAD	Signature Activation Data is a set of data involved in the signature activation protocol (SAP), which are used to authenticate and authorise the signature completion operation in the TOE. D.SAD consists of: 1. D.Reference_Signer_Authentication_Data, 2. D.Authorisation_Data A part of the D.SAD (D.SAD / D.application SignaturePart) is created on TSE side using D.PIN, so it is the knowledge base part of the secret indirectly.	confidentiality, integrity

Name	Description	Security
D.SCD	Signature Creation Data. In the conventional digital signature systems, this corresponds to the private key of the key pair. In the Smart-ID system, the D.SCD is never generated or combined in the single location, instead the three components of the D.SCD (D.clientPart, D.serverPart, D.serverShare) are generated and processed within distinct sub-systems.	confidentiality, integrity
D.server SignaturePart	Share of the signature of D.DTBS/R, which is computed by the TOE with the D.serverPart. It is not possible to validate the D.serverSignaturePart with any public key.	confidentiality, integrity
D.server Signature Share	Share of the signature of D.DTBS/R, which is created with the private key D.serverShare. The D.serverSignatureShare can be validated with the D.serverModulus.	confidentiality, integrity, nonrepudiation
D.serverModulus	Data, which can certify the integrity of D.server SignatureShare. This is the public part of the D.serverShare/D.serverModulus key pair.	integrity
D.serverPart	Part of the D.SCD of the Signer. Server part of the private key, generated in the TSE and transmitted to TOE and protected by the TOE.	confidentiality, integrity
D.serverShare	Part of the D.SCD of the Signer. Server share of the private key, generated and protected by the HSM.	confidentiality, integrity
D.signature	Signature of the D.DTBS/R, which is created with the private key corresponding to the compound of D.clientPart and D.serverPart and D.serverShare. Such kind of private key does not exist, therefore the signature is instead created from the signature shares D.serverSignatureShare and D.application SignatureShare. The D.signature can be validated with the D.SVD.	integrity, nonrepudiation
D.Signer	Set of data, which represents the Signer and his/her identity. In SecureZone S.Signer is represented by D.Signing_Key_Id	integrity, authenticity
D.Signing_ Key_Id	The signing key is the private key of an asymmetric key pair used to create a digital signature under the signer's sole control. The signing key in the Smart-ID system is D.SCD. The TOE uses the asset D.Signing_Key_Id associated to D.clientPart, D.serverPart in the database and D.serverShare in the Cryptographic Module. D.Signing_Key_Id is referenced as keyUUID in some places since this is the name of this attribute in the developer documents and sources.	integrity

Name	Description	Security
D.SVD	Signature verification data is the public part, associated with the signing key, to perform digital signature verification. In Smart-ID system it is the compoundModulus created by D.clientModulus and D.serverModulus. Data, which can certify the integrity of the D.signature. The integrity of the D.SVD is protected by the certificate issued by the CA.	integrity
D.TEK	Symmetric cryptographic key shared between the TOE and specific instance of TSE. D.TEK is established during the key pair enrolment with the Diffie-Hellman key exchange algorithm. It is used to protect the communication between the TSE instance and SecureZone.	confidentiality, integrity
D.TSF_ CONFIG_ DATA	It is the set of TOE configuration data used to operate the TOE. It contains the following data: • D.OTP • D.KTK • D.KWK • D.TEK	confidentiality, integrity
D.VC	A short representation of the D.DTBS/R, for example, four first digits of the digest of the D.DTBS/R. The D.VC is computed by SCA and TSE. SecureZone does not compute or process the D.VC.	integrity

4.2 Subjects

The TOE provides services and functions to the following external entities (natural persons and external IT systems) and uses the following list of subjects and roles in order to regulate access to the assets.

4.2.1 Natural Persons

1. U.User – Registered user of the Smart-ID services. U.User is using the TOE services to produce Qualified Electronic Signatures. U.User owns the mobile device with the Smart-ID App installed on it. Smart-ID App provides convenient user interface for the TOE services. Depending on the level of authentication (multi-factor or single-factor), the U.User is either bound to the subject S.Signer or to the subject S.App.
2. U.Admin – Administrator of the Smart-ID SecureZone, who installs, configures and maintains the TOE. Note that even though the TOE is installed, configured and administrated by the Administrator, the authentication of the Administrator is handled

by the supporting IT environment, for example, by the operating system, which the TOE is running on top of and the HSM module, by the use of OCS password.

The TOE and TOE environment's administrative functions that involve installation and operation of the HSM module, shall be conducted at least under dual control.

TOE allows Administrator to use the following functions:

a. supply the OCS password to the TOE.
b. generate the new batch of D.serverShare assets, which will be used during the new key-pair enrolments.

4.2.2 External IT Systems

The following external IT systems use the services and functions of the TOE:

1. U.Monitoring – The IT component in the environment, which is quering the TOE status, health and monitoring information. This information is public and is provided to the monitoring component without authentication and access control. The U.Monitoring is bound to the subject S.Anonymous when processing the queries.
2. U.CA – The IT component "Smart-ID CA", which is managing the certificates. The U.CA is bound to the subject S.CA after the authentication of requests. U.CA executes the following functions:

   a. destroying of the key-pair after the revocation of the certificate,
   b. starting the re-key process of the key-pair.

4.2.3 Subjects

TOE uses the following list of subjects when processing the requests and performing the access control decisions to the functions and assets.

1. S.Signer – Owner of the D.SCD, who is using the TOE functions to produce Qualified Electronic Signatures. U.User is bound to the S.Signer after the successful multi-factor authentication, which includes the possession-based information (from the mobile device) and the knowledge-based information, which only the U.User knows.
2. S.App - The Smart-ID App instance in the mobile device of the U.User. The Smart-ID App is using the technical TOE functions (such as update D.OTP, perform re-key operation) on behalf of the U.User, the S.App has limited access to the TOE objects. U.User is bound to the S.App after the successful single-factor authentication, which includes the possession-based information from the mobile device.
3. Privileged users:

   3.1 S.Admin – Subject S.Admin is used when administrators perform the management functions of the TOE and they authenticate themselves with the OCS password. Because the sensitive data fields in the TOE database are encrypted, administrators cannot modify them without supplying the valid OCS password. In that sense, the HSM is providing the authentication function for the administrators. The TOE and TOE environment's administrative functions that involve installation and operation of the HSM module, shall be conducted at least under dual control.

   3.2 S.CA – The IT component Smart-ID CA (U.CA) is bound to the S.CA after trusted channel-based authentication which is configured by S.Admin.

4. S.Anonymous – This subject is used, when the access control to the TOE services is handled with other environment measures, such as network firewalls and other measures, which do not provide the personalised identification. For example, the TOE status, health and monitoring information and some key pair status information is provided to other components within the larger PKI system, without personalised authentication and the requesting user is not fully known.

4.2.4 Roles

TOE uses the following list of roles, when processing the request and deciding the access control:

1. R.Signer - The role R.Signer is used only when the U.User has been authenticated with multi-factor authentication.
2. R.App - The role R.App is used when the Smart-ID App is using the technical TOE functions (such as update D.OTP, perform re-key operation) on behalf of the U.User.
3. R.Admin – The role R.Admin is used when the subject S.Admin is authenticated with the HSM OCS password and the administrative function is performed. Note that the TOE and TOE environment's administrative functions that involve installation and operation of the HSM module, must be conducted at least under dual control.
4. R.CA – The role R.CA is used when the IT component Smart-ID CA (U.CA) is authenticated and starts the key destroying or the re-key function.
5. R.Anonymous - The role R.Anonymous is used when the user is bound to the S.Anonymous.

The described roles are only logical entities, the mapping between the subjects and roles are hard-coded in the TOE configuration and source code. TOE doesn't need to implement the administrative, dynamic role and permissions management.

4.3 Threat Agents

1. S.Attacker – A human or process acting on his behalf, located outside of the TOE. It is assumed that S.Attacker has complete knowledge about the components of the Smart-ID system, the structure of the TOE, algorithms, and API interfaces. However, he doesn't know any secret values, e.g. the key material. S.Attacker has high attack potential.

4.4 Threats

The following kind of threats are considered within this ST document. The main goal of the S.Attacker is to perform one of the following sub-attacks:

1. create one or more forged D.signatures of fresh D.DTBS/R under the name of Signer or
2. decrease the trust in the signatures created with the service Smart-ID Trust Service Provider (TSP) and in the security of the TOE.

ST document organises the individual threats in subsections, in order to present closely related threats next to each other.

4.4.1 Threats related to the key enrolment

Attacker may use the vulnerabilities of the key enrolment process to impersonate the Signer or to derive the D.SCD of the Signer or get the Signer's certificate issued for the different key pair. The following specific threats are considered within this ST document.

4.4.1.1 T.Enrolment_Signer_Authentication_Data_Disclosed

An attacker is able to obtain whole or part of D.Reference_Signer_Authentication_Data during enrolment. This can be during generation, storage or transfer of the data to the TOE or transfer between signer and TOE. As an example it could be:

- by eavesdropping on the TSSP key enrolment run and retrieving the components of D.Reference_Signer_Authentication_Data, which are transmitted from the Signer to the TOE.

Such data disclosure may allow a potential incorrect Signer authentication leading to unauthorised signature operation on behalf of Signer (Signer's signature on the fresh D.DTBS/R without Signer's consent).

> Application Note 1
>
> There is no separate processes for the authentication of the Signer and for the signing key activation in the TOE. The Signer authentication and the signature completion happens together according to the section 2.3.3 – Signature generation process, and the process steps from 9 to 13 using the same asset components.

4.4.1.2 T.Enrolment_Signer_Impersonation

Attacker impersonates signer during enrolment. As an example, it could be:

- performing MITM attack on the TSSP key enrolment run and modifying the value of the Signer's key pair, such as D.SCD components. Attacker may then use the modified values to forge the signatures of the Signer or to impersonate the Signer to the TOE, in order to create Signer's signature on the fresh D.DTBS/R without Signer's consent.

The asset D.Reference_Signer_Authentication_Data is threatened.

This is the same threat as T.ENROLMENT_SIGNER_IMPERSONATION in PP 419 241-2 [6].

4.4.1.3 T.SVD_Forgery

Attacker modifies the D.SVD during transmission to the RA or CA. This results in loss of integrity in the binding of D.SVD to signing key and to D.Signer.

The asset D.SVD is threatened.

If the CA relies on the generation of the key pair controlled by the TOE as specified in ETSI 319 411-1 [15], clause 6.3.3 d) then an attacker can forge signatures masquerading as the signer.

This is the same threat as T.SVD_Forgery in PP 419 241-2 [6].

Application Note 2

Issuing the certificate verifies the CSR – “proof of possession or control of the private key”, associated with the D.SVD, as specified in ETSI 319 411-1 [15], clause 6.3.1 a). Therefore, this threat is countered without any specific measures within the TOE.

4.4.1.4 T.Random

Attacker guesses system secrets and is able to create or modify TOE objects or participate in communication with external systems.

D.Random is used to generate the D.SCD and other encryption/decryption keys. If attacker is able to guess random numbers, the attacker may be able to successfully derive the value of the D.SCD or other encryption/decryption keys and then impersonate the Signer to the TOE or create Signer’s signature on the fresh D.DTBS/R without Signer’s consent.

The asset D.Random is threatened.

This is the same threat as T.Random in PP 419 241-2 [6].

4.4.2 Threats related to impersonation of the Signer within the signing process

Attacker may use the vulnerabilities in the signing process and try to impersonate the Signer to the TOE or use some other ways to get TOE to create Signer’s signature without the Signer’s consent. The following specific threats are considered within this ST document.

This group of threats correspond to a more general threat T.SigF_Misuse from PP 14169-2 [16].

4.4.2.1 T.SAD_Forgery

Attacker forges or manipulates D.SAD during transfer in TSSP and is able to create a signature on the fresh D.DTBS/R without the Signer having authorised the operation.

The asset D.SAD is threatened.

4.4.2.2 T.SAP_ByPass

Attacker bypasses one or more steps in the TSSP and is able to create a signature without the signer having authorised the operation.

The asset D.SAD is threatened.

4.4.2.3 T.SAP_Replay

Attacker replays one or more steps of TSSP and is able to create a signature on the fresh D.DTBS/R without the signer having authorised the operation.

The asset D.SAD is threatened.

4.4.2.4 T.TSSP_Modification

Attacker modifies the user’s data and/or security attributes within the TOE data storage and is able to submit the query to the TOE’s signing function so that TOE outputs the Signer’s signature on the fresh D.DTBS/R without the Signer’s consent.

The assets D.Reference_Signer_Authentication_Data and D.Signing_Key_Id are threatened.

This threat corresponds to the threats T.MAINTENACE_AUTHENTICATION_DISCLOSE and T.SIGNER_AUTH_DATA_MODIFIED as in PP 419 241-2 [6].

4.4.2.5 T.TSSP_Duplication

Attacker gets hold of the D.clientPart, D.OTP and D.TEK and impersonates Signer to the TOE's signing function. Attacker is able to submit the valid D.applicationSignatureShare with the fresh D.DTBS/R so that TOE outputs the Signer's signature on the fresh D.DTBS/R without the Signer's consent.

The assets D.clientPart, D.OTP, D.TEK and D.applicationSignatureShare are threatened.

This threat corresponds to the T.AUTHENTICATION_SIGNER_IMPERSONATION in the PP 419 241-2 [6].

4.4.3 Threats related to signature forgery

Attacker may use the vulnerabilities in the cryptographic algorithm and the signature scheme itself or the hashing function itself and try to claim that Signer has signed such documents, which he has not intended. The following specific threats are considered within this ST document.

4.4.3.1 T.Signature_Forgery

Attacker uses the vulnerability in the cryptographic signature algorithm and without having the copy of the D.SCD, forges the value of the new signature for the fresh D.DTBS/R, which can be successfully validated with D.SVD.

The asset D.signature is threatened.

4.4.3.2 T.DTBSR_Forgery

Attacker modifies the D.DTBS/R, before it is submitted to the Signer from the SCA (Signature Creation Application) or within the TOE, during the execution of the TOE's signing function. Attacker can then get the signature on a different kind of D.DTBS/R than was intended to be signed by the Signer.

The asset D.DTBS/R is threatened.

This threat corresponds to the T.DTBSR_Forgery in the PP 419 241-2 [6].

4.4.4 Other threats

Attacker may use other attacks on the TOE to create the signatures and he may also try to attack the audit log of the TOE in order to claim that Signer has signed some documents, which he has not intended. The following specific threats are considered within this ST document.

4.4.4.1 T.Admin_Impersonation

Attacker impersonates a Privileged User and updates D.Reference_Signer_Authentication_Data, D.Signing_Key_Id and/or D.SVD. Such data modification may allow a potential incorrect signer authentication leading to unauthorised signature operation on behalf of Signer.

The assets D.Reference_Signer_Authentication_Data, D.Signing_Key_Id, D.Authorisation_Data, D.Reference_App_Authentication_Data and D.SVD are threatened.

This threat corresponds to the threats T.ADMIN_IMPERSONATION, T.AUTHORISATION_DATA_UPDATE and T.AUTHORISATION_DATA _DISCLOSE as in PP 419 241-2 [6].

4.4.4.2 T.Privileged_User_Insertion

Attacker is able to create D.Privileged_User including D.Reference_Privileged_User_Authentication_Data and is able to log on to the TOE as a Privileged User.

The assets D.Privileged_User and D.Reference_Privileged_User_Authentication_Data are threatened.

4.4.4.3 T.Reference_Privileged_User_Authentication_Data_Modification

An attacker modifies D.Reference_Privileged_User_Authentication_Data and is able to log on to the TOE as the Privileged User.

The asset D.Reference_Privileged_User_Authentication_Data is threatened.

4.4.4.4 T.Audit_Alteration

Attacker modifies system audit and is able hide trace of TOE modification or usage on the following ways:

- Attacker attacks the audit function of the TOE or the audit log storage outside of the TOE and deletes the existing log entries, modifies the existing log entries or creates new log entries. Attacker is then able to hide his own actions and attack attempts or he is able to claim that the Signer has signed a different kind of D.DTBS/R than intended by the Signer, even though the corresponding D.signature may not even exist.

The asset D.Audit_Data is threatened.

4.4.4.5 T.Context_Alteration

Attacker modifies system configuration D.TSF_CONFIG_DATA to perform an unauthorised operation on the following ways:

- Attacker gets the root-level or physical access to the TOE and is able modify the user's data, security attributes and program code of the TOE and is able to produce the D.signature for the fresh D.DTBS/R, which the Signer has not intended to sign.

The asset: D.Signing_Key_Id, D.SAD, D.SVD, D.Reference_Signer_Authentication_Data, D.Reference_App_Authentication_Data and D.TSF_CONFIG_DATA are threatened.

4.4.4.6 T.Signature_Request_Disclosure

Attacker obtains knowledge of D.DTBS/R or D.SAD during transfer to TOE.

The assets D.DTBS/R and D.SAD are threatened.

If the D.DTBS/R and D.SAD do not require confidentiality, then this threat is mitigated.

4.4.5 Relations between threats and assets

Table 4. Compiled overview of relations between threats and assets

Asset	Security Requirement	Threats
D.Signing_Key_Id	Integrity	T.TSSP_Modification, T.Admin_Impersonation, T.Context_Alteration
D.serverShare	Integrity, Confidentiality	
D.serverPart	Integrity, Confidentiality	

Table 4. Compiled overview of relations between threats and assets

Asset	Security Requirement	Threats
D.SCD	(virtual asset)	
D.PIN	Confidentiality	
D.clientPart	Integrity, Confidentiality	T.TSSP_Duplication
D.Authorisation_Data	Integrity, Confidentiality	T.Admin_Impersonation
D.DTBS	Integrity	
D.DTBS/R	Integrity	T.DTBSR_Forgery, T.Signature_Request_Disclosure
D.VC	Integrity	
D.SAD	Integrity, Confidentiality	T.SAD_Forgery, T.SAP_Bypass, T.SAP_Replay, T.Context_Alteration, T.Signature_Request_Disclosure
D.applicationSignaturePart	Integrity, Confidentiality	
D.serverSignaturePart	Integrity, Confidentiality	
D.applicationSignature Share	Integrity, Confidentiality, Non-repudiation	T.TSSP_Duplication
D.serverSignatureShare	Integrity, Confidentiality, Non-repudiation	
D.signature	Integrity, Non-repudiation	T.Signature_Forgery
D.SVD	Integrity	T.SVD_Forgery, T.Admin_Impersonation, T.Context_Alteration
D.clientModulus	Integrity	
D.serverModulus	Integrity	
D.Audit_Data	Integrity	T.Audit_Alteration
D.Signer	Integrity, Confidentiality	

Table 4. Compiled overview of relations between threats and assets

Asset	Security Requirement	Threats
D.Reference_Signer_Authentication_Data	Integrity	T.Enrolment_Signer_Authentication_Data_Disclosed, T.Enrolment_Signer_Impersonation, T.TSSP_Modification, T.Admin_Impersonation, T.Context_Alteration
D.Reference_App_Authentication_Data	Integrity, Confidentiality	T.Admin_Impersonation, T.Context_Alteration
D.TSF_CONFIG_DATA	Integrity, Confidentiality	T.Context_Alteration
D.Privileged_User	Integrity	T.Privileged_User_Insertion
D.Reference_Privileged_User_Authentication_Data	Integrity, Confidentiality	T.Privileged_User_Insertion, T.Reference_Privileged_User_Authentication_Data_Modification
D.Random	Integrity, Confidentiality	T.Random
D.DEK	Integrity, Confidentiality	
D.TEK	Integrity, Confidentiality	T.TSSP_Duplication
D.KWK	Integrity, Confidentiality	
D.KTK	Integrity, Confidentiality	
D.OTP	Integrity, Confidentiality	T.TSSP_Duplication

4.5 Organization Security Policies

4.5.1 P.SCD_Confidential

The confidentiality of D.SCD must be reasonably assured (from reg. (EU) 910/2014 [4], Annex II, point 1.(a)).

4.5.2 P.SCD_Unique

Any given instance of a D.SCD shall occur only once (from reg. (EU) 910/2014 [4], Annex II, point 1.(b)).

4.5.3 P.Sig_unForgeable

An electronic signature shall be reliably protected against forgery using currently available technology. It shall not be possible, with reasonable assurance, to derive an electronic signature from data other than the D.SCD (from reg. (EU) 910/2014 [4], Annex II, point 1.(c)).

4.5.4 P.SCD_userOnly

D.SCD of a legitimate Signer shall be reliably protected against use by others (from reg. (EU) 910/2014 [4], Annex II, point 1.(d) and Article 26, point (c)).

4.5.5 P.DTBS_Integrity

The TOE and its environment shall not alter D.DTBS nor D.DTBS/R. The TOE and its environment shall not prevent such data from being presented to the Signer prior to signing (from reg. (EU) 910/2014 [4], Annex II, point 2).

4.5.6 P.TSP_Qualified

Generating or managing D.SCD on behalf of the Signer may only be done by a qualified trust service provider (from reg. (EU) 910/2014 [4], Annex II, point 3).

4.5.7 P.SCD_Backup

The TSP may duplicate the D.SCD only for back-up purposes provided the 1) security of the duplicated datasets must be at the same level as for the original datasets and 2) number of duplicated datasets shall not exceed the minimum needed to ensure continuity of the service (from reg. (EU) 910/2014 [4], Annex II, point 4).

4.5.8 P.TSP_QCert

The TSP must use a trustworthy Certificate Generation Application (CGA) to generate a qualified certificate for the SVD generated by TOE. The TSP must ensure that the advanced electronic signature is uniquely linked to the Signer and the Signer can be identified through the related certificate (from reg. (EU) 910/2014 [4], Article 26, point (a) and (b)).

4.5.9 P.DTBS/R_Unique

The electronic signature must be linked to D.DTBS in such a way that any subsequent change in data is detectable – for example, any subsequent change in data shall result in a different D.DTBS/R generated for this data (from reg. (EU) 910/2014 [4], Article 26, point (d)).

4.5.10 P.Reliable_Audit

The TSP shall keep reliable audit records about the signing events.

4.6 Assumptions

4.6.1 A.CA

It is assumed that the qualified TSP that issues qualified certificates is compliant with the relevant requirements for qualified TSP's as defined in reg. (EU) 910/2014 [4]. The CGA

protects the authenticity of the Signer's name and the SVD in the qualified certificate by an advanced electronic signature of the CSP.

4.6.2 A.ACCESS_PROTECTED

The TOE environment limits physical and logical access to the components in the TOE environment to authorized U.Admin. The TOE software and hardware environment is maintained by U.Admin in a secure state, including protection against unauthorized software and configuration changes. The TOE environment provides reasonable protection against denial of service attacks.

It is assumed that copies of data protected by the TOE are managed outside of the TOE, provides appropriate protection for that data to a level required by the application context and the risks in the deployment environment.

Informative: based on Application note 21 of the PP 419 241-2 [6] the following data are managed outside the TOE:

- D.clientPart
- D.DTBS/R
- D.applicationSignaturePart
- D.Audit_Data
- D.serverShare

4.6.3 A.PRIVILEGED_USER

The U.Admin, who has unrestricted physical and logical access to the TOE and the TOE environment, is well-trained, trusted and performs his duties. The U.Admin is trusted. The TOE and TOE environment's administrative functions that involve installation and operation of the HSM module, shall be conducted at least under dual control.

4.6.4 A.SIGNER_ENROLMENT

The signer shall be enrolled and certificates managed in conformance with the regulations given in reg. (EU) 910/2014 [4]. Guidance for how to implement an enrolment and certificate management system in conformance with eIDAS reg. (EU) 910/2014 [4] are given in e.g. ETSI 319 411-1 [15] or for qualified certificate in e.g. ETSI 319 411-2 [17].

4.6.5 A.SIGNER_AUTHENTICATION_DATA_PROTECTION

It is assumed that the signer will not disclose his authentication factors.

4.6.6 A.SIGNER_DEVICE

The Signer has the trusted TSE component in his environment to help him to complete the TSSP steps for key generation and signing operations. The TSE component is evaluated with the EAL2 level, according to the ST document [9] and it fulfils the security objectives OE.TSE.*.

4.6.7 A.TSP_AUDITED

It is assumed that the TSP deploying the TOE is a qualified TSP and audited to be compliant with the requirements for TSP's given by reg. (EU) 910/2014 [4].

4.6.8 A.CSPRNG

It is assumed that the HSM provides the secure random number generator, which can be used by the TOE to generate the cryptographic keys and random nonces. It is required that the random number generator satisfies the statistical tests from the suite FIPS 140-2 [14].

4.6.9 A.CRYPTO

It is assumed that cryptographic algorithms, algorithm parameters and key lengths, which are in use by TOE, are endorsed by recognized authorities as appropriate, for the use of TSPs. This includes algorithms for generating of random numbers and signing key pairs and algorithms for creating signatures as well as the algorithms for protecting integrity and confidentiality of TOE assets.

> Application Note 3
>
> TOE supports the standard cryptographic algorithms and recommended key sizes according to ETSI TS 119 312 [18] and [19].

4.6.10 A.JVM

It is assumed that the TOE is the only application, which is running on the JVM.

5 Security Objectives (ASE_OBJ)

This chapter identifies and defines the security objectives for the TOE and its environment. Objectives counter the identified threats and comply with the organizational security policies and assumptions.

5.1 Security Objectives for the TOE

5.1.1 OT.SCD_Confidential

TOE shall keep the D.serverPart components of the D.SCD confidential.

5.1.2 OT.Sig_Secure

TOE shall generate electronic signatures, that cannot be forged without knowledge of the D.SCD, through robust cryptographic techniques. The TOE shall not allow the D.SCD to be reconstructed from the digital signatures.

5.1.3 OT.SCD/SVD_Corresp

TOE shall ensure the correspondence between the D.SVD and the D.SCD. This includes unambiguous reference of a created SVD/SCD pair for export of the D.SVD and in creating a digital signature with the D.SCD.

5.1.4 OT.TSSP_End2End

TOE shall protect the confidentiality and integrity of the communications between the TOE and with the Signer. The TOE shall not allow the attacker to eavesdrop and modify the information transmitted between the TOE and the Signer.

5.1.5 OT.SAP_Replay_Protection

TOE shall protect the communications between the TOE and with the Signer against the replay attack.

5.1.6 OT.TSSP_Require_clientSignatureShare

TOE shall protect the signature creation function of the TOE by following the TSSP and requiring the valid D.applicationSignatureShare in order to create the D.signature.

5.1.7 OT.TSSP_Validate_clientSignatureShare

TOE shall protect the signature creation function of the TOE by following the TSSP and validating the D.applicationSignatureShare in order to make sure that the correct D.clientPart has been used to create the D.applicationSignatureShare (validated with the D.clientModulus).

5.1.8 OT.TSSP_CloneDetection

TOE shall protect the signature creation function of the TOE by following the TSSP and detecting the usage of incorrect D.OTP in signature creation requests with valid D.application SignatureShare. This situation indicates that Signer's local environment has been cloned. The valid D.clientPart has leaked, but only one of the clients has been issued the correct D.OTP for the subsequent key pair operation. TOE shall initiate revocation of the Signer's certificate and destroy the respective key pair after detecting such situation.

5.1.9 OT.TSSP_TimeDelay_Locks

TOE shall protect the signature creation function of the TOE by following the TSSP and after submission of incorrect D.applicationSignatureShare (which most likely indicates that the Signer has entered the wrong D.PIN to the TSE), TOE shall prevent the immediate re-try of the signature creation request with new D.applicationSignatureShare for the same D.DTBS/R. The TOE shall apply time-delay between accepting the new requests and shall initiate revocation of the Signer's certificate and destroy the respective key pair after the limit of incorrect D.applicationSignatureShare has been reached.

5.1.10 OT.DTBS/R_Protect

TOE shall protect the D.DTBS/R from substitution and modification. The protection shall be applied, when the D.DTBS/R is transmitted from or to another IT component from the TOE environment.

5.1.11 OT.Audit_Events

TOE shall create audit records about the important system events.

5.1.12 OT.Privileged_User_Management

The TOE shall ensure that any modification to D.Privileged_User and D.Reference_Privileged_User_Authentication_Data are performed under control of a Privileged User.

5.1.13 OT.Privileged_User_Authentication

The TOE shall ensure that an administrator as a Privileged User is authenticated before any action on the TOE is performed.

> Application Note 4
>
> The exception to this objective is when the initial (set of) Privileged Users are created as part of system initialisation.

5.1.14 OT.Privileged_User_Protection

The TOE shall ensure that data associated to D.Privileged_User are protected in integrity and if needed in confidentiality.

5.2 Security Objectives for the Environment fulfilled by HSM

The HSM inside the TOE environment is CC evaluated and conforming to the QSCD requirements. This means that the HSM fulfils several Security Objectives by design. Because HSM processes the components of the D.SCD and provides important security functions to the TOE, it is useful to show, which security objectives for the environment are fulfilled by the HSM itself.

5.2.1 OE.HSM.SCD_Confidential

The HSM shall protect the confidentiality of the components of the D.SCD.

5.2.2 OE.HSM.SCD_Unique

The HSM shall ensure cryptographic quality of generated keys. HSM shall generate the D.serverShare (component of the D.SCD) and the corresponding D.serverModulus (component of the D.SCD) securely. It shall not be possible to derive D.serverShare from D.serverModulus and probability of equal D.serverShares shall be negligible.

5.2.3 OE.HSM.Sig_Secure

The HSM shall generate electronic signatures (D.serverSignatureShare) that cannot be forged without knowledge of the private key (D.serverShare), through robust cryptographic techniques. The D.serverShare cannot be reconstructed from the digital signatures.

5.2.4 OE.HSM.Tamper_Resistance

The HSM shall prevent or resist physical tampering with HSM device and components.

5.2.5 OE.HSM.Sigy_SigF

The HSM shall provide the share of the signature (D.serverSignatureShare) creation function for the TOE only and protects the D.serverShare against attempts by other users to create a digital signature using it.

5.2.6 OE.HSM.DTBS/R_Integrity

The HSM shall ensure that the D.DTBS/R cannot be altered when processed by the HSM.

5.3 Security Objectives for the Environment fulfilled by TSE

The Threshold Signature Engine (TSE) inside the Signer environment is CC evaluated and it fulfils security objectives by design. Because TSE processes the components of the D.SCD and provides important security functions to the Signer, it is useful to show, which security objectives for the environment are fulfilled by the TSE itself.

5.3.1 OE.TSE.Sig_Secure

The TSE shall generate D.applicationSignaturePart, that cannot be forged without knowledge of the D.clientPart, through robust cryptographic techniques. The TSE shall not allow the private key to be reconstructed from the digital signatures.

5.3.2 OE.TSE.SCD_Unique

The TSE shall ensure cryptographic quality of generated keys. TSE shall generate the D.clientShare (component of the D.SCD) and the corresponding D.clientModulus (component of the D.SVD) securely. It shall not be possible to derive D.clientShare from D.clientModulus and probability of equal D.clientShares shall be negligible.

5.3.3 OE.TSE.SCD_Confidential

The TSE shall protect the confidentiality of the components of the D.SCD.

5.3.4 OE.TSE.TSSP_End2End

The TSE shall protect the confidentiality and integrity of the communications between the TSE and TOE.

5.3.5 OE.TSE.DTBS_Intend

The TSE shall allow verification of the integrity of the D.DTBS/R, so that the Signer can be sure he is signing the same document he intends to sign.

5.3.6 OE.TSE.App_Sandbox

The TSE shall be run in isolated mobile app process, protected from other apps.

5.4 Security Objectives for the Environment fulfilled by other components

5.4.1 OE.CA_REQUEST_CERTIFICATE

The operational environment shall ensure that the qualified TSP that issues qualified certificates is compliant with the relevant requirements for qualified TSP's as defined in reg. (EU) 910/2014 [4].

The operational environment shall use a process for requesting a certificate, including D.SVD and signer information, and CA signature in a way, which demonstrates the signer is in control of the signing key associated with the D.SVD presented for certification. The integrity of the request shall be protected.

5.4.2 OE.Env

The TSP deploying the TOE is a qualified TSP and audited to be compliant with the requirements for TSP's given by reg. (EU) 910/2014 [4]. The audit of the qualified TSP shall cover the security objectives for the operational environment specified in this clause.

The TOE environment shall provide a J2EE application server which ensures that the TOE is the only application deployed in a container of the J2EE application server.

The TOE environment shall limit physical and logical access to the components in the TOE environment to authorised U.Admin. The TOE software, hardware environment and backup datasets shall be maintained by U.Admin in a secure state, including protection against unauthorised software and configuration changes.

5.4.3 OE.Trusted_Timestamps

The TOE environment shall provide trusted timestamps.

5.4.4 OE.TrustedAdmin

The U.Admin, who has unrestricted physical and logical access to the TOE and the TOE environment shall be well-trained and trusted and shall perform his duties.

5.4.5 OE.SVD_AUTHENTICITY

The operational environment shall ensure the D.SVD integrity during transmit outside the TOE to the CA.

The TOE environment shall ensure the integrity of the D.SVD exported by the TOE to the CGA. The CGA shall verify the correspondence between the D.SCD of the Signer and the D.SVD in the input provided to the certificate generation function of the CGA.

5.4.6 OE.DTBS_Intend

The Signature Creation Application (SCA) generates the D.DTBS/R of the data that has been presented as D.DTBS, which the Signer intends to sign. The TOE environment shall allow verification of the integrity of the D.DTBS/R, so that the Signer can be sure he is signing the same document he intends to sign.

5.4.7 OE.DTBS/R_Protect

The TOE environment shall ensure that the D.DTBS/R cannot be altered in transit between physically separated components of the TOE environment.

5.4.8 OE.DTBS/R_Unique

The TOE environment shall ensure that D.DTBS may practically have only one unique representation as D.DTBS/R. TOE environment shall ensure that the probability for existane of two different D.DTBS-s having identical D.DTBS/R is negligible.

5.4.9 OE.CGA_QCert

The CGA shall generate qualified certificate and thus confirm that the D.SCD, corresponding to the certified D.SVD, is under the control of Signer. The CGA shall include identifying information of the Signer in the certificate and therefore enable to identify the Signer by the signature.

5.4.10 OE.Protected_AuditLog

The TOE environment shall protect the integrity of the audit log and protect the audit log from unauthorized deletion.

5.4.11 OE.CSPRNG

The HSM must provide the cryptographically secure random number generator for the TOE. TOE will use the RNG provided to generate D.OTP, D.TEK and D.DEK.

> Application Note 5
>
> The environment objective OE.CSPRNG has been defined to accurately reflect the implementation, where the SZ is using the HSM-provided random number generation service and it is not implementing the random number generation on its own.

5.4.12 OE.Signer_Authentication_Data

The signer's management of authentication factors data outside the TOE shall be carried out in a secure manner.

5.5 Security Objectives Rationale

5.5.1 Mapping between SPD and Security Objectives

The mapping between Security Problem Definition (SPD) and security objectives has been divided into multiple tables for size considerations, according to the type of the security objectives:

1. mapping to TOE security objectives is shown in the table 5 on page 55,
2. mapping to HSM security objectives is shown in the table 6 on page 56,
3. mapping to TSE security objectives is shown in the table 7 on page 57,
4. mapping to general environment security objectives is shown in the table 8 on page 58.

Table 5. Mapping between Security Problem Definition (SPD) and TOE security objectives

	OT.SCD_Confidential	OT.Sig_Secure	OT.TSSP_End2End	OT.SAP_Replay_Protection	OT.TSSP_Require_clientSignatureShare	OT.TSSP_Validate_clientSignatureShare	OT.TSSP_CloneDetection	OT.TSSP_TimeDelay_Locks	OT.DTBS/R_Protect	OT.SCD/SVD_Corresp	OT.Audit_Events	OT.Privileged_User_Management	OT.Privileged_User_Authentication	OT.Privileged_User_Protection
T.Enrolment_Signer_ Authentication_Data_ Disclosed			X											
T.Enrolment_Signer_ Impersonation			X											
T.SAD_Forgery						X		X						
T.SAP_ByPass					X									
T.SAP_Replay				X										
T.TSSP_Modification										X				
T.TSSP_Duplication							X							
T.Signature_Forgery		X												
T.DTBSR_Forgery				X					X					

Table 5. Mapping between Security Problem Definition (SPD) and TOE security objectives

	OT.SCD_Confidential	OT.Sig_Secure	OT.TSSP_End2End	OT.SAP_Replay_Protection	OT.TSSP_Require_clientSignatureShare	OT.TSSP_Validate_clientSignatureShare	OT.TSSP_CloneDetection	OT.TSSP_TimeDelay_Locks	OT.DTBS/R_Protect	OT.SCD/SVD_Corresp	OT.Audit_Events	OT.Privileged_User_Management	OT.Privileged_User_Authentication	OT.Privileged_User_Protection
T.Admin_Impersonation					X									
T.Context_Alteration					X									
T.Signature_Request_ Disclosure				X	X	X								
T.Privileged_User_ Insertion												X	X	
T.Reference_Privileged_ User_Authentication_ Data_Modification												X	X	X
P.SCD_Confidential	X		X											
P.Sig_unForgeable		X												
P.SCD_userOnly	X		X	X	X	X	X	X						
P.DTBS_Integrity			X						X					
P.Reliable_Audit											X			

Table 6. Mapping between Security Problem Definition (SPD) and HSM security objectives

	OE.HSM.SCD_Confidential	OE.HSM.SCD_Unique	OE.HSM.Sig_Secure	OE.HSM.Tamper_Resistance	OE.HSM.Sigy_SigF	OE.HSM.DTBS/R_Integrity
T.Random		X		X		

Table 6. Mapping between Security Problem Definition (SPD) and HSM security objectives

	OE.HSM.SCD_Confidential	OE.HSM.SCD_Unique	OE.HSM.Sig_Secure	OE.HSM.Tamper_Resistance	OE.HSM.Sigy_SigF	OE.HSM.DTBS/R_Integrity
T.Signature_Forgery			X			
T.DTBSR_Forgery						X
T.Context_Alteration				X		
T.Signature_Request_ Disclosure				X		
P.DTBS_Integrity						X
P.SCD_Confidential	X			X		
P.SCD_Unique		X				
P.Sig_unForgeable			X			
P.SCD_userOnly	X	X		X	X	
A.Crypto		X	X			

Table 7. Mapping between Security Problem Definition (SPD) and TSE security objectives

	OE.TSE.Sig_Secure	OE.TSE.SCD_Unique	OE.TSE.TSSP_End2End	OE.TSE.DTBS_Intend	OE.TSE.SCD_Confidential	OE.TSE.App_Sandbox
T.Enrolment_Signer_Authentication_Data_Disclosed			X			
T.Enrolment_Signer_Impersonation			X			
T.Signature_Forgery	X					
T.DTBSR_Forgery				X		
T.Random		X				
P.SCD_Unique		X				
P.SCD_Confidential			X		X	X
P.SCD_userOnly		X	X		X	X

Table 7. Mapping between Security Problem Definition (SPD) and TSE security objectives

	OE.TSE.Sig_Secure	OE.TSE.SCD_Unique	OE.TSE.TSSP_End2End	OE.TSE.DTBS_Intend	OE.TSE.SCD_Confidential	OE.TSE.App_Sandbox
P.DTBS_Integrity				X		
P.Sig_unForgeable	X					
A.SIGNER_DEVICE	X	X	X	X	X	X

Table 8. Mapping between Security Problem Definition (SPD) and environment security objectives

	OE.Env	OE.SVD.Authenticity	OE.DTBS_Intend	OE.DTBSR/R_Protect	OE.DTBS/R_Unique	OE.CGA_QCert	OE.TrustedAdmin	OE.Trusted_Timestamps	OE.Protected_AuditLog	OE.CSPRNG	OE.CA_REQUEST_CERTIFICATE	OE.Signer_Authentication_Data
T.TSSP_Modification						X						
T.Context_Alteration	X											
T.SVD_Forgery		X				X					X	
T.DTBSR_Forgery			X	X								
T.Audit_Alteration									X			
P.DTBS_Integrity			X	X								
P.DTBS/R_Unique					X							
P.TSP_QCert						X						
P.SCD_Backup	X											
P.TSP_Qualified	X											
P.Reliable_Audit								X	X			
A.ACCESS_PROTECTED	X											
A.CA		X				X					X	
A.PRIVILEGED_USER							X					
A.SIGNER_ENROLMENT	X											

Table 8. Mapping between Security Problem Definition (SPD) and environment security objectives

	OE.Env	OE.SVD.Authenticity	OE.DTBS_Intend	OE.DTBSR/R_Protect	OE.DTBS/R_Unique	OE.CGA_QCert	OE.TrustedAdmin	OE.Trusted_Timestamps	OE.Protected_AuditLog	OE.CSPRNG	OE.CA_REQUEST_CERTIFICATE	OE.Signer_Authentication_Data
A.SIGNER_AUTHENTICATION_DATA_PROTECTION												X
A.TSP_AUDITED	X											
A.CSPRNG										X		
A.CRYPTO										X		
A.JVM	X											

5.5.2 Security Objectives Rationale

5.5.2.1 Rationale for mitigating threats

5.5.2.1.1 Mitigating T.Enrolment_Signer_Authentication_Data_Disclosed and T.Enrolment_Signer_Impersonation

T.Enrolment_Signer_Authentication_Data_Disclosed (Attacker eavesdrops on the TSSP key enrolment run and retrieves the D.SCD components, which are transmitted from the Signer to the TOE) and T.Enrolment_Signer_Impersonation (Attacker performs the MITM attack on the TSSP key enrolment run and modifies the value of the Signer's key pair, such as D.SCD components) is mitigated by OT.TSSP_End2End and OE.TSE.TSSP_End2End, which in combination, give the following assurances:

1. Signer authenticates the TOE by the known public key.
2. TOE authenticates the instance of the Signer by the Signer's Diffie-Hellman public key and the shared symmetric encryption key.
3. Signer and the TOE use the Diffie-Hellman key exchange protocol to create the shared symmetric encryption key to protect the confidentiality and integrity of communication channel.

5.5.2.1.2 Mitigating T.SAD_Forgery

T.SAD_Forgery (Attacker submits forged value of D.applicationSignatureShare to the TOE's signing function and is able to get the Signer's signature on the fresh D.DTBS/R) is mitigated by the OT.TSSP_Validate_clientSignatureShare and OT.TSSP_TimeDelay_Locks.

First, OT.TSSP_Validate_clientSignatureShare ensures that the TOE computes the D.serverSignaturePart and combines it with the submitted D.applicationSignaturePart and creates the D.applicationSignatureShare. The validity of the D.applicationSignatureShare is

verified with the D.clientModulus. Because only Signer has the correct D.clientPart, which was required to create the D.applicationSignaturePart, the TOE shall prevent the Signer impersonation and shall provide the signature creation function for the legitimate Signer only.

Secondly, OT.TSSP_TimeDelay_Locks limits the number of tries the attacker has to guess the correct D.PIN or D.clientPart. The TOE security objective OT.TSSP_TimeDelay_Locks ensures that TOE doesn't immediately accept new signature creation try for the same D.DTBS/R. It also ensures that TOE will destroy the key pair and initiate the revocation of the respective certificate after the limit of incorrect signature creation tries has been reached.

5.5.2.1.3 Mitigating T.SAP_ByPass

T.SAP_ByPass (Attacker bypasses the access control part of the TOE's signing function and is able to get the Signer's signature on the fresh D.DTBS/R without providing the valid D.applicationSignatureShare) is mitigated by the OT.TSSP_Require_clientSignatureShare.

This security objective ensures that TOE implements the TSSP correctly and computes the compound signature D.signature only when the valid D.applicationSignatureShare is available. In fact, the cryptographic properties of the TSSP ensure that the computed D.signature is only valid, in case all the signature shares, which are used (D.applicationSignatureShare and D.serverSignatureShare), are valid as well.

5.5.2.1.4 Mitigating T.SAP_Replay

T.SAP_Replay (Attacker eavesdrops the data, which is submitted to the TOE's signing function by the Signer and is able to modify the data and replay it, so that the TOE outputs the Signer's signature on the fresh D.DTBS/R without the Signer's consent) is mitigated by OT.SAP_Replay_Protection.

This security objective ensures that TOE implements the TSSP correctly and computes the D.serverSignaturePart on the submitted D.DTBS/R and combines it with the submitted D.applicationSignaturePart and in turn, creates the D.applicationSignatureShare on the submitted D.DTBS/R. In fact, the D.applicationSignatureShare is the RSA signature and it has the cryptographic properties that in case the signed message has been changed, the signature is not valid anymore. Therefore, it is not possible to change the D.DTBS/R, after the Signer created the D.applicationSignaturePart for the particular D.DTBS/R.

5.5.2.1.5 Mitigating T.TSSP_Modification

T.TSSP_Modification (Attacker modifies the user's data and/or security attributes within the TOE data storage and is able to submit the query to the TOE's signing function so that TOE outputs the Signer's signature on the fresh D.DTBS/R) is mitigated by OT.SCD/SVD_Corresp and OE.CGA_QCert.

First, the TOE security objective OT.SCD/SVD_Corresp ensures that D.SCD and D.SVD is corresponding to each other in cryptographic means. So, in case the attacker is able to modify some components of the D.SCD, the compound D.SCD and the D.SVD are no longer corresponding to each other.

Secondly, the environment security objective OE.CGA_QCert ensures that the authentic value of the D.SVD is recorded in the certificate issued by the CA. In case the attacker is able to modify the D.SCD, the authentic value of the public key (D.SVD from the certificate) is no longer corresponding to the modified private key.

5.5.2.1.6 Mitigating T.TSSP_Duplication

T.TSSP_Duplication (attacker gets hold of the D.clientPart, D.OTP and D.TEK and impersonates Signer to the TOE's signing function) is mitigated by OT.TSSP_CloneDetection.

The TOE security objective OT.TSSP_CloneDetection ensures that TOE detects the situations, when the valid D.applicationSignatureShare is submitted to the signature creation function, but with the old or incorrect D.OTP. This indicates that multiple clients have been operating and only one of the clients has been issued the correct D.OTP for the subsequent key pair operation. In this case, the key pair is destroyed and the respective certificate's revocation is initiated by the TOE.

5.5.2.1.7 Mitigating T.Signature_Forgery

T.Signature_Forgery (attacker uses the vulnerability in the cryptographic signature algorithm and without having the copy of the D.SCD, crafts the value of the new signature for the fresh D.DTBS/R) is mitigated by OT.Sig_Secure, OE.HSM.Sig_Secure and OE.TSE.Sig_Secure.

First, the TSE security objective OE.TSE.Sig_Secure ensures that it is not possible to generate the D.applicationSignaturePart without access to the private key D.clientPart, by ensuring that the TSE performs the signature computation according to the RSA signature algorithm and with using specified key sizes.

Secondly, the HSM security objective OE.HSM.Sig_Secure ensures that it is not possible to generate the D.serverSignatureShare without access to the private key D.serverShare, by ensuring that the HSM performs the signature computation according to the RSA signature algorithm and with using specified key sizes.

Finally, the TOE security objective OT.Sig_Secure ensures that it is not possible to generate the D.serverSignaturePart without access to the private key D.serverPart and finally, that it is not possible to generate the compound signature D.signature without having access to the components of the D.SCD (D.clientPart, D.serverPart and D.serverShare). This is ensured by the TOE by performing the signature computation according to the RSA signature algorithm and TSSP and with using specified key sizes.

Therefore, signature forgery without having access to the D.SCD is not possible.

5.5.2.1.8 Mitigating T.DTBSR_Forgery

T.DTBSR_Forgery (attacker modifies the D.DTBS/R before or during the signing process) is mitigated by following security objectives.

1. OT.SAP_Replay_Protection – Attacker cannot submit the eavesdropped signature creation request with the modified D.DTBS/R, because the D.applicationSignaturePart is depending on the D.DTBS/R and cannot be validated anymore. Therefore, the replay is not possible.
2. OT.DTBS/R_Protect – D.DTBS/R is protected, when TOE is processing the signature creation request or transmitting the D.DTBS/R to another IT component.
3. OE.DTBS/R_Protect – D.DTBS/R is protected by the environment, when the signature creation request is submitted from the SCA.
4. OE.HSM.DTBS/R_Integrity – D.DTBS/R is protected, when HSM is processing the request to create the D.serverSignatureShare.
5. OE.DTBS_Intended and OE.TSE.DTBS_Intended – SCA computes the D.VC from D.DTBS/R and displays to the Signer. TSE also computes the D.VC from the D.DTBS/R, which the Signer is about to sign and displays to the Signer. Singer can then compare the two VC-s and understand, what is the context of the signing operation and verify that

it is the correct DTBS. Signer would then be notified if the attacker managed to change the D.DTBS/R when transmitted from SCA.

Therefore, the combination of the security objectives prevent the substitution of D.DTBS/R.

5.5.2.1.9 Mitigating T.Admin_Impersonation

T.Admin_Impersonation (attacker personates the privileged user of the TOE and executes the TOE's signing function for the Signer) is mitigated by OT.TSSP_Require_clientSignatureShare.

Fulfilling the security objective OT.TSSP_Require_clientSignatureShare means that even when the attacker manages to execute the TOE internal functions directly, he cannot create the D.signature for the fresh D.DTBS/R without the corresponding D.applicationSignaturePart, which can only be created with the D.clientPart, under control of the Signer.

5.5.2.1.10 Mitigating T.Privileged_User_Insertion

T.Privileged_User_Insertion (Attacker is able to create D.Privileged_User including D.Reference_Privileged_User_Authentication_Data and is able to log on to the TOE as a Privileged User) is covered by OT.Privileged_User_Management requiring only Privileged User can create new R.Privileged_User and OT.PRIVILEGED_USER_AUTHENTICATION that requires a Privileged User to be authenticated.

5.5.2.1.11 Mitigating T.Reference_Privileged_User_Authentication_Data_Modification

T.Reference_Privileged_User_Authentication_Data_Modification (an attacker modifies D.Reference_Privileged_User_Authentication_Data and is able to log on to the TOE as the Privileged User) is covered by OT.PRIVILEGED_USER_MANAGEMENT requiring only Privileged User can modify R.Privileged_User and OT.PRIVILEGED_USER_AUTHENTICATION that requires a Privileged User to be authenticated.

It is also covered by OT.PRIVILEGED_USER_PROTECTION requiring the Privileged User to be protected in integrity.

5.5.2.1.12 Mitigating T.Audit_Alteration

T.Audit_Alteration (attacker attacks the audit function of the TOE or the audit log storage outside of the TOE and deletes the existing log entries, modifies the existing log entries or creates new log entires) is mitigated by OE.Protected_Auditlog, which ensures that TOE environment protects the audit records.

5.5.2.1.13 Mitigating T.Context_Alteration

T.Context_Alteration (attacker gets the root-level or physical access to the TOE or depending IT components and is able modify the user's data, security attributes and program code) is mitigated by following security objectives.

1. OE.Env – The environment of the TOE provides the first-level protection against physical attacks.
2. OE.HSM.Tamper_Resistance – The HSM security objective provides protection against physical attacks and provides resistance to the tampering with the security attributes and program code protected by HSM. Because D.serverSignatureShare can only be created by HSM and D.serverSignatureShare is required to create the D.signature, the tamper resistance is extended to the D.signature as well.

3. OT.TSSP_Require_clientSignatureShare – The TOE security objective means that even in case the attacker has managed to break other security features, the D.application SignaturePart is still required according to the TSSP. The D.applicationSignaturePart can only be created with the D.clientPart, under control of the Signer.

Therefore, the combination of the abovementioned security objectives prevent the physical attack.

5.5.2.1.14 Mitigating T.Signature_Request_Disclosure

T.Signature_Request_Disclosure Attacker obtains knowledge of R.DTBS/R or R.SAD during transfer to TOE) is mitigated by following security objectives.

1. OT.SAP_Replay_Protection – The environment of the TOE provides the first-level protection against physical attacks.
2. OT.TSSP_Require_clientSignatureShare – The security objective provides protect the signature creation function of the TOE by following the TSSP and requiring the valid D.applicationSignatureShare in order to create the D.signature.
3. OE.HSM.Tamper_Resistance – The security objective provides protection against physical attacks and provides resistance to the tampering with the security attributes and program code protected by HSM. Because D.serverSignatureShare can only be created by HSM and D.serverSignatureShare is required to create the D.signature, the tamper resistance is extended to the D.signature as well.
4. OT.TSSP_Validate_clientSignatureShare – The TOE security objective means that even in case the attacker has managed to break other security features, the D.application SignaturePart is still required according to the TSSP. The D.applicationSignaturePart can only be created with the D.clientPart, under control of the Signer.

Therefore, the combination of the abovementioned security objectives prevent the physical attack.

5.5.2.1.15 Mitigating T.Random

Threat T.Random (attacker guesses the random values, which are used to generate the D.SCD and is able to successfully derive the value of the D.SCD) is mitigated by following security objectives.

1. OE.TSE.SCD_Unique - The TSE security objective ensures the cryptographic quality of generated keys. This includes the cryptographic quality random number generator.
2. OE.HSM.SCD_Unique - The HSM security objective ensures the cryptographic quality of generated keys. This includes the cryptographic quality random number generator.
3. OE.HSM.Tamper_Resistance - The HSM security objective ensures that the HSM internal random number generator cannot be influenced by attacker.

The combination of security objectives ensure that attacker cannot guess random values.

5.5.2.1.16 Mitigating T.SVD_Forgery

Threat T.SVD_Forgery (attacker modifies the D.SVD value, which is created by TOE and presented to the CA for the certification of the Signer's key pair) is mitigated by OE.SVD_ AUTHENTICITY, OE.CA_REQUEST_CERTIFICATE and OE.CGA_QCert.

The environment objective OE.SVD_AUTHENTICITY and OE.CA_REQUEST_CERTIFICATE ensure the integrity of the SVD exported by the TOE to the CGA. The environment objective OE.CGA_QCert ensures that the CA verifies that the Signer has control over the D.SCD corresponding to the D.SVD presented for certification.

5.5.2.2 Rationale for fulfilling organisational policy requirements

5.5.2.2.1 Fulfilling P.SCD_Confidential

P.SCD_Confidential (The confidentiality of SCD must be reasonably assured) is addressed by following objectives:

1. OT.SCD_Confidential
2. OT.TSSP_End2End
3. OE.TSE.App_Sandbox
4. OE.TSE.TSSP_End2End
5. OE.TSE.SCD_Confidential
6. OE.HSM.SCD_Confidential
7. OE.HSM.Tamper_Resistance

In the Smart-ID system, the D.SCD consists of the three shares residing in physically separated components. In order to export D.SCD outside the TOE environment, an attacker needs to be able to export and successfully decrypt all of the three shares together. The confidentiality of the corresponding shares of the D.SCD is assured as shown in the table 9, by securing them in transit, at rest and when in use.

Table 9. Protection of the components of the D.SCD

Component	Protection assurances		
	Data in transit	Data at rest	Data in use
D.clientPart	D.clientPart is not transmitted anywhere	D.clientPart is stored inside the mobile app sandbox, encrypted with the key derived from VAD. The OE.TSE.SCD_Confidential is defined in [9].	D.clientPart is generated securely, inside the mobile app process, isolated from other apps. SCD.clientPart is used securely, inside the mobile app process, isolated from other apps. The OE.TSE.App_Sandbox is defined in [9].

Table 9. Protection of the components of the D.SCD

Component	Protection assurances		
	Data in transit	Data at rest	Data in use
D.serverPart	Transmitted over protected communication channel into the TOE. When in transmission, the D.serverPart is encrypted with the D.TEK. Refer to OE.TSE.TSSP_End2End and OT.TSSP_End2End.	Stored in the TOE database, wrapped with D.KWK. The OT.SCD_Confidential is assuring the confidentiality of this operation.	D.serverPart is generated securely, inside the mobile app process, isolated from other apps. D.serverPart is used securely, inside the SecureZone process. The OT.SCD_Confidential is assuring the confidentiality of this operation.
D.serverShare	Not transmitted anywhere.	Stored in the TOE database, wrapped with HSM encryption (with HSM master key). The OE.HSM.SCD_Confidential and OE.HSM.Tamper_Resistance is assuring the confidentiality of this operation.	D.serverShare is generated and processed in clear only in the HSM. The OE.HSM.SCD_Confidential and OE.HSM.Tamper_Resistance is assuring the confidentiality of this operation.

5.5.2.2.2 Fulfilling P.Sig_unForgeable

P.Sig_unForgeable (electronic signature shall be reliably protected against forgery and it shall not be possible, to derive an electronic signature from data other than the D.SCD) is addressed by OT.Sig_Secure, OE.HSM.Sig_Secure and OE.TSE.Sig_Secure, in a same way as the threat T.Signature_Forgery (attacker uses the vulnerability in the cryptographic signature algorithm and without having the copy of the D.SCD, crafts the value of the new signature for the fresh D.DTBS/R) is mitigated. Refer to the corresponding section about mitigating T.Signature_Forgery.

5.5.2.2.3 Fulfilling P.SCD_userOnly

P.SCD_userOnly (D.SCD shall be reliably protected against use by others) is addressed by the mitigation of the following threats:

1. T.Enrolment_Signer_Authentication_Data_Disclosed
2. T.Enrolment_Signer_Impersonation
3. T.SAD_Forgery
4. T.SAP_ByPass
5. T.SAP_Replay
6. T.TSSP_Modification

7. T.TSSP_Duplication
8. T.Admin_Impersonation
9. T.Context_Alteration
10. T.Random

All those threats impact the policy requirement that the D.SCD shall be reliably protected against use by others than the legitimate Signer. Refer to the individual sections about the mitigation of those threats. In summary, they are mitigated by the following security objectives:

1. OT.SCD_Confidential
2. OT.TSSP_End2End
3. OT.SAP_Replay_Protection
4. OT.TSSP_Require_clientSignatureShare
5. OT.TSSP_Validate_clientSignatureShare
6. OT.TSSP_CloneDetection
7. OT.TSSP_TimeDelay_Locks
8. OE.TSE.TSSP_End2End
9. OE.TSE.SCD_Unique
10. OE.TSE.SCD_Confidential
11. OE.TSE.App_Sandbox
12. OE.HSM.SCD_Unique
13. OE.HSM.SCD_Confidential
14. OE.HSM.Tamper_Resistance
15. OE.HSM.Sigy_SigF

5.5.2.2.4 Fulfilling P.DTBS_Integrity

P.DTBS_Integrity (the TOE and its environment shall not alter DTBS nor DTBS/R and not prevent such data from being presented to the Signer prior to signing) is addressed by the following security objectives:

1. OT.TSSP_End2End ensures that when D.DTBS/R is transmitted from TSE to TOE, the transmission is encrypted and cannot be changed.
2. OT.DTBS/R_Protect ensures that when D.DTBS/R is processed in TOE or transmitted to another IT components D.DTBS/R is protected from substitution and modification.
3. OE.HSM.DTBS/R_Integrity ensures that D.DTBS/R is protected when processed by HSM.
4. OE.DTBS_Intend and OE.TSE.DTBS_Intend ensures that Signer can verify the integrity of the D.DTBS/R and Signer can be sure that he is signing the correct DTBS.
5. OE.DTBS/R_Protect ensures that D.DTBS/R is protected when transmitted in the TOE environment.

5.5.2.2.5 Fulfilling P.SCD_Unique

P.SCD_Unique (any given instance of a SCD shall occur only once) is addressed by OE.HSM.SCD_Unique and OE.TSE.SCD_Unique. In the Smart-ID system, the D.SCD consists of the three shares residing in a physically separated components.

The D.clientPart and D.serverPart is generated in the TSE. The OE.TSE.SCD_Unique ensures that TSE generates cryptographic quality D.clientShare/D.clientModulus key pair and that probability of the equal D.clientShare is negligible.

The D.serverShare is generated in the HSM. The OE.HSM.SCD_Unique ensures that HSM generates cryptographic quality D.serverShare/D.serverModulus key pair and that probability of the equal D.serverShare is negligible.

5.5.2.2.6 Fulfilling P.DTBS/R_Unique

P.DTBS/R_Unique (the electronic signature must be linked to D.DTBS in such a way that any subsequent change in data is detectable) is addressed by OE.DTBS/R_Unique, which by the use of appropriate cryptographic techniques ensures that generating such data, which would match a given D.DTBS/R is infeasible, thus ensuring that the signed data and the electronic signature are securely linked together. Any subsequent change in data will result in a different D.DTBS/R and is therefore detectable.

5.5.2.2.7 Fulfilling P.TSP_Qualified

P.TSP_Qualified (generating or managing the SCD may only be done by a qulified trust service provider) is addressed by OE.Env, which ensures that the TSP is audited.

5.5.2.2.8 Fulfilling P.TSP_QCert

P.TSP_QCert (the TSP must use a trustworthy CGA to generate a qualified certificate for the SVD generated by Smart-ID) is addressed by OE.CGA_QCert which ensures that the CGA generates a qualified certificate and thus confirms with the generated certificate that the SCD, corresponding to the certified SVD, is under the control of U.Signer. Signatures created by the U.Signer are uniquely linked to the U.Signer and it is possible to identify the U.Signer by the signature.

5.5.2.2.9 Fulfilling P.Reliable_Audit

P.Reliable_Audit (the TOE shall keep reliable audit records about events in the TOE) is addressed by combination of following objectives:

1. OT.Audit_Events - ensures that audit records will be generated about the important system events.
2. OE.Protected_AuditLog - ensures that audit records are reliably timestamped and protected from modifications.
3. OE.Trusted_Timestamps - ensures that TOE can use the operating system provided trusted timestamps.

5.5.2.2.10 Fulfilling P.SCD_Backup

P.SCD_Backup (the security of backups must be at the same level as for the original datasets) is addressed by OE.Env, which ensures that TSP secures the backups and keeps the datasets at minimum.

5.5.2.3 Rationale for fulfilling assumptions

5.5.2.3.1 Fulfilling A.CA

A.CA (the CGA protects the authenticity of the Signer's name and the SVD in the qualified certificate by an advanced electronic signature of the TSP) is addressed by OE.SVD_AUTHENTICITY, OE.CA_REQUEST_CERTIFICATE and OE.CGA_QCert. The OE.SVD_AUTHENTICITY ensures integrity of the SVD exported by the TOE to the CGA. OE.CA_REQUEST_CERTIFICATE ensures that the integrity of the request of the certificate including D.SVD and signer information is protected. OE.CGA_QCert ensures that the CGA generates a qualified certificate and thus confirms with the generated certificate that the SCD, corresponding to the certified SVD, is under the control of U.Signer. Signatures created by the U.Signer are uniquely linked to the U.Signer and it is possible to identify the U.Signer by the signature.

5.5.2.3.2 Fulfilling A.ACCESS_PROTECTED

A.ACCESS_PROTECTED (the TOE environment limits physical and logical access to the components in the TOE environment) is addressed by OE.Env which ensures that the TOE environment is protected and limits the exposure to physical attacks.

5.5.2.3.3 Fulfilling A.PRIVILEGED_USER

A.PRIVILEGED_USER (the U.Admin is trusted) addressed by OE.TrustedAdmin, which ensures that the U.Admin is well trained and trusted to perform his duties.

5.5.2.3.4 Fulfilling A.SIGNER_ENROLLMENT

A.SIGNER_ENROLLMENT (the signer enrolment is conformant with reg. (EU) 910/2014 [4]) addressed by OE.Env, which ensures that the TSP is audited.

5.5.2.3.5 Fulfilling A.SIGNER_AUTHENTICATION_DATA_PROTECTION

A.SIGNER_AUTHENTICATION_DATA_PROTECTION (the signer will not disclose his authentication factors) addressed by OE.Signer_Authentication_Data, which ensures that signer's management of authentication factors data outside the TOE is carried out in a secure manner.

5.5.2.3.6 Fulfilling A.SIGNER_DEVICE

A.SIGNER_DEVICE (Signer has the trusted and evaluated TSE component in his environment to help him to complete the TSSP steps for key generation and signing operations) is adressed by environment objectives marked OE.TSE.*.

5.5.2.3.7 Fulfilling A.TSP_AUDITED

A.TSP_AUDITED (TSP deploying the TOE is a qualified TSP) is addressed by OE.Env which ensures that the TOE operator is a qualified TSP.

5.5.2.3.8 Fulfilling A.CSPRNG

A.CSPRNG (HSM provides the secure random number generator) is fulfilled by OE.CSPRNG which provides a cryptographically secure random number generator.

5.5.2.3.9 Fulfilling A.CRYPTO

A.CRYPTO (endorsed algorithms, algorithm parameters and key lengths) is fulfilled by OE.CSPRNG, which provides a cryptographically secure random number generator and by OE.HSM.SCD_Unique and OE.HSM.Sig_Secure.

5.5.2.3.10 Fulfilling A.JVM

A.JVM (TOE is the only application running on the JVM) is fulfilled by OE.ENV, which ensures that the TOE is the only application deployed in the container included in J2EE application server.

6 Extended components definition (ASE_ECD)

There is no extended components used in SZ.

7 Security Requirements (ASE_REQ)

7.1 Data in TOE: user data and TSF data

This section classifies the assets defined in the ASE_SPD and security attributes used in the SFR definitions.

7.1.1 User data

Those attributes are considered 'user data' as per the definition of the CC Part 2, page 21, paragraph 36. These are the attributes, which TOE places no special meaning and doesn't use them for any security related functions.

The protection of user data is handled by the access control policies defined in SFRs FDP_ACC.1 and FDP_ACF.1.

Table 10. User data attributes in the TOE

Attribute name	Corresponding asset	Storage location	Notes
DTBSR	D.DTBS/R	in memory only	The digest for the signing. Submitted by the client during the performSignature() operation (see also table 13).

7.1.2 TSF data

Rest of the data handled by TOE is classified as 'TSF data' as per the definition of the CC Part 2, page 21, paragraph 36.

7.1.2.1 Authentication data

Following attributes in the table 11 are considered 'authentication data' as per the definition of the CC Part 2, page 21, paragraph 40. Authentication data is used to verify the claimed identity of a user requesting services from a TOE. Authentication data is used by the authentication mechanisms defined in SFRs FIA_UAU.3 and FIA_UAU.5. The authentication data itself is protected with the SFRs from the family FPT and FMT.

Table 11. Authentication data attributes in the TOE

Attribute name	Corresponding asset	Storage location	Notes
client_share_2nd_part	D.serverPart	database, wrapped	This is the other half of the D.clientShare. It is used to complete the signature share D.applicationSignatureShare.
client_modulus	D.clientModulus	database	This is the public key of the D.clientShare key pair. It is used to verify the signature share D.applicationSignatureShare.
server_modulus	D.serverModulus	database	Generated by the HSM and stored in the TOE database
composite_modulus	D.SVD	database	Computed by the TOE and stored in the TOE database
current_one_time_password	D.OTP	database	This is the next one-time-password, which is expected to be sent by TSE for the next key pair operation. D.OTP is not wrapped but is rather stored in hashed form. This value is only used in comparison.
sz_keypair_uuid	D.Signing_Key_Id	database	This is the identifier for the key pair.

7.1.2.2 Security data

Following attributes are considered 'security attributes' as per the definition of the CC Part 2, page 21, paragraph 35. Security attributes are used by TSF in order to make decisions as required by the SFRs. Security attributes are protected with the SFRs from the family FPT and FMT.

Table 12. Security attributes in the TOE

Attribute name	Corresponding asset	Storage location	Notes
DEK_symmetric_key	D.DEK	database, wrapped	This is used by TOE to encrypt and to integrity protect some database fields. D.DEK is generated and used by TOE itself, and it is wrapped with the D.KTK.
server_privatekey	D.serverShare	database, wrapped	Generated by the HSM and stored in the TOE database, wrapped with the HSM master key.

Table 12. Security attributes in the TOE

Attribute name	Corresponding asset	Storage location	Notes
TEK_symmetric_key	D.TEK	database, wrapped	This is generated by TOE during the Diffie-Hellman key exchange and is afterwards used to encrypt/decrypt the messages transmitted between TSE and TOE. HMAC portion of the key is used to provide and verify the authenticity and integrity of the messages. It is actually wrapped, by using the D.DEK.
KTK_wrapper_key	D.KTK	database, wrapped	This is used by TOE to sign the replies to the initateKey() operation and to allow the TSE to authenticate the TOE.
KWK_wrapper_key	D.KWK	database, wrapped	This is used by TOE to wrap the key material in the TOE database.
sz_keypair_state		database	The status of the key pair, for example 'IN_PREPARATION', 'READY', 'TIMELOCKED'.
locked_until_time		database	Timestamp until the key is not usable.
pin_attempts		database	Number of times the FIA_UAU.5.2/Signer authentication method has failed in a row.
expiration_time		database	Timestamp after which the key is not usable.
DH_keyPair		ephemeral, in memory	Temporary DH key pair, which is used to generate the D.TEK. After the D.TEK is established and stored, the DH_keyPair is destroyed.

7.2 Security Function Policies (SFP)

This section defines the rules for the access control decisions performed by the TSF and referenced from the SFR definitions.

The SFPs are defined in the tabular form. The tables are processed from up to down. In case the request parameters match with the attributes in the row of table, the corresponding

access control decision is looked up. In case none of the previous rows matched the request, the last line is usually the wildcard match, with the access control decision to deny the request.

7.2.1 Operations

First of all, the overview is provided of the operations, which can be requested by users and admins. Those operations correspond to the TOE API methods and further information, including the detailed list of method arguments and error conditions, can be found in the architecture documents. The table 13 gives the short summary about the user operations and table 14 lists the admin operations.

Table 13. List of operations, which can be requested by TOE users

Operation name	Description
initiateKey	This is the first method to be called by TSE in order to enrol new key pair with the TOE. The method establishes the D.TEK and also D.OTP, which is used in the subsequent methods.
submitClient2ndPart	This is the second method to be called by TSE during the new key pair enrolment.
performSignature	This is the main method to create signatures with the enrolled key pair. TSE submits the digest to be signed, the signature part computed in the Signer's environment and other multi-factor authentication data. TSE receives the completed signature.
reKey	This is the method to complete the re-key process, which is initiated by the CA in order to initiate generation of new server share of the private key and the corresponding new compound public key for the Signer.
refreshCloneDetection	This is the technical method used by TSE to request the fresh D.OTP without creating any signature.
getKeyState	This is the technical method used by TSE to get the status information about the key pair, for example, the remaining time until the key pair is un-locked.
getFreshnessToken	This is the technical method to ensure the key pair operations are performed in sequence on different cluster nodes and they do not conflict with each other.
revokeKey	This method is used by TSE and the CA to destroy the key pair in the TOE so that it cannot be used anymore.

Table 14. List of operations, which can be requested by TOE admins

Operation name	Description
hsmPasswordEntry	This method is used by the admin after starting the TOE, in order to load the HSM password. The HSM password is not stored in the configuration file and must be entered on each boot manually.

Table 14. List of operations, which can be requested by TOE admins

Operation name	Description
batchGenerateServerShares	This method is used by the admin to pre-generate D.serverShare assets, so that Signer enrolment can be done quicker.
generateKTKKey	This method is used by the admin to generate D.KTK.
generateKWKKey	This method is used by the admin to generate D.KWK.
generateDEKKey	This method is used by the admin to generate D.DEK.

7.2.2 SFP/Init

Table 15. Security Function Policy, which specifies the default values for the new attributes and objects created by the TOE.

Object or attribute	Operation	Default value
sz_keypair_state	initiateKey	'IN_PREPARATION'
pin_attempts	initiateKey	0
expiration_time	initiateKey	current time + 3 years

7.2.3 SFP/Signer

The SFP/Signer is regulating the access to the signature generation function of the TOE. Only Signer should have access to this function and only after he has authenticated himself with knowledge-based and possession-based authentication factors.

The "objects related to authenticated D.Signing_Key_Id" mean all the database fields, which are associated with the same D.Signing_Key_Id as the Signer identifier. Essentially, "objects owned by authenticated Signer".

Table 16. Security Function Policy, which specifies when the U.User is allowed to perform the operation performSignature.

User	Subject	Role	Objects	Operation	Rule
U.User	S.Signer	R.Signer	objects related to authenticated D.Signing_Key_Id	perform-Signature	allow
U.User	S.Signer	R.Signer	*	perform-Signature	deny
U.User	S.Signer	R.Signer	*	*	deny

In the table below, it is further specified which TSF data attributes the authenticated R.Signer can manage in the course of the allowed performSignature operation.

Table 17. TSF data attributes managed by the R.Signer.

Operation	change_default	query	modify	delete
performSignature	-	D.serverShare, D.serverModulus, D.SVD	-	-

7.2.4 SFP/App

The SFP/App is regulating access to technical functions of the TOE. TSE uses those functions on behalf of the Signer and uses only possession-based authentication factors to authenticate himself.

Table 18. Security Function Policy, which specifies what are the access rights of the S.App.

User	Subject	Role	Objects	Operation	Rule
U.User	S.App	R.App	objects related to authenticated D.Signing_Key_Id	submitClient-2ndPart	allow
U.User	S.App	R.App	objects related to authenticated D.Signing_Key_Id	reKey	allow
U.User	S.App	R.App	objects related to authenticated D.Signing_Key_Id	refreshClone-Detection	allow
U.User	S.App	R.App	other objects	submitClient-2ndPart, reKey, refreshClone-Detection	deny
U.User	S.App	R.App	*	*	deny

In the table below, it is further specified which TSF data attributes the authenticated R.App can manage in the course of the allowed operations.

Table 19. TSF data attributes managed by the R.App.

Operation	change_default	query	modify	delete
submitClient-2ndPart	-	D.Signing_Key_Id, D.OTP	D.serverPart, D.OTP	-
reKey	-	D.Signing_Key_Id, D.OTP	D.serverShare, D.serverModulus, D.SVD, D.OTP	D.serverShare, D.serverModulus, D.OTP
refreshClone-Detection	-	D.Signing_Key_Id, D.OTP	D.OTP	-

Table 19. TSF data attributes managed by the R.App.

Operation	change_default	query	modify	delete

7.2.5 SFP/Anonymous

The SFP/Anonymous is regulating access to technical functions of the TOE, which do not require personalised user identification/authentication and strict access control. For example, all users are permitted to enrol new key pair and all users are permitted to query status of the key pair and get freshness tokens. Also, destroying of the key pair is not authenticated, because user may not have control of the authentication factors anymore.

This doesn't mean that the access to those methods is wide open without any security. The other components of the Smart-ID system and network devices are configured to perform the preliminary access control and the channel-based authentication is still performed by those components and devices.

The "New object with fresh D.Signing_Key_Id" means that new keyUUID is generated, which doesn't match with any existing keyUUIDs. Essentially, "the new object, which will be owned by the new Signer, who made the request".

Table 20. Security Function Policy, which specifies what are the access rights of the un-authenticated users.

User	Subject	Role	Objects	Operation	Rule
N/A	S.Anonymous	R.Anonymous	new object with fresh D.Signing_Key_Id	initiateKey	allow
N/A	S.Anonymous	R.Anonymous	attributes 'lockDurationSec', 'pinAttemptsLeft', 'wrongAttempts', 'status' of the object of the requested D.Signing_Key_Id	getKeyState	allow
N/A	S.Anonymous	R.Anonymous	attribute 'freshnessToken' of the object of the requested D.Signing_Key_Id	getFreshness-Token	allow
N/A	S.Anonymous	R.Anonymous	attribute 'status' of the object of requested D.Signing_Key_Id	revokeKey	allow
N/A	S.Anonymous	R.Anonymous	*	*	deny

7.2.6 SFP/Admin

The SFP/Admin is regulating access to the admins.

The "new object D.serverShare not associated with any existing D.Signing_Key_Id" means that administrator can only request the generation of the new and fresh D.serverShares and

cannot access any such D.serverShare values, which are already "in use" by some existing key pair.

Table 21. Security Function Policy, which specifies what are the access rights of the admins.

User	Subject	Role	Objects	Operation	Rule
U.Admin	S.Admin	R.Admin	in-memory OCS password	hsmPassword-Entry	allow
U.Admin	S.Admin	R.Admin	new object D.serverShare not associated with any existing D.Signing_Key_Id	batchGenerate-ServerShares	allow
U.Admin	S.Admin	R.Admin	new object D.KTK	generateKTKKe	allow
U.Admin	S.Admin	R.Admin	new object D.KWK	generateKWKKe	allow
U.Admin	S.Admin	R.Admin	new object D.DEK	generateDEKKe	allow
U.Admin	S.Admin	R.Admin	*	*	deny

In the table below, it is further specified which TSF data attributes the authenticated R.Admin can manage in the course of the allowed operations.

Table 22. TSF data attributes managed by the R.Admin.

Operation	change_default	query	modify	delete
hsmPassword-Entry	-	-	-	-
batchGenerate-ServerShares	-	-	D.serverShare	-
generateKTK-Key	-	-	D.KTK	-
generateKWK-Key	-	-	D.KWK	-
generateDEK-Key	-	-	D.DEK	-

7.2.7 SFP/CA

The SFP/CA is regulating access to the administrative functions, which are required by the CA. CA can call the prepareReKey and revokeKey operations on any existing key pairs.

Table 23. Security Function Policy, which specifies what are the access rights of the CA.

User	Subject	Role	Objects	Operation	Rule
U.CA	S.CA	R.CA	requested D.Signing_Key_Id	prepare-ReKey	allow
U.CA	S.CA	R.CA	requested D.Signing_Key_Id	revokeKey	allow
U.CA	S.CA	R.CA	*	*	deny

In the table below, it is further specified which TSF data attributes the authenticated R.CA can manage in the course of the allowed operations.

Table 24. TSF data attributes managed by the R.CA.

Operation	change_default	query	modify	delete
prepareReKey	-	D.Signing_Key_Id	-	-
revokeKey	-	D.Signing_Key_Id	-	D.serverShare, D.serverModulus, D.SVD, D.OTP

7.3 Security Functional Requirements

This document uses the following typograhic conventions, as suggested in the https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Interpretationen/AIS_41_BSI_PP_ST_Guide_pdf.pdf?__blob=publicationFile:

- Iterations of the SFRs are denoted by a slash "/" and the iteration indicator after the component, for example FCS_CKM.1/RSA_SVD.
- Refinements of security requirements made by the ST author are denoted in such a way that added words are in **bold, highlighted text** and removed words are ~~strikethrough~~.
- Selections having been made by the ST author are denoted as *italic, highlighted text* and in addition a footnote will show the original text from [2].
- Assignments having been made by the ST author are denoted in the same way as selections.

7.3.1 Security Audit (FAU)

7.3.1.1 Security audit generation (FAU_GEN.1)

7.3.1.1.1 FAU_GEN.1 – Security audit generation

FAU_GEN.1.1	The TSF shall be able to generate an audit record of the following auditable events: a) Start-up and shutdown of the audit functions; b) All auditable events for the *not specified*[a] level of audit; and c) *Other specifically defined auditable events:*[b] *1) Privileged User authentication;* *2) Signer management;* *3) Signer authentication;* *4) Signing key generation;* *5) Signing key destruction;* *6) Signing key activation and usage, including the D.DTBS/R and the hash of D.signature;*

[a] selection, choose one of: minimum, basic, detailed, not specified [b] assignment: other specifically defined auditable events

Application Note 6

The PP 419 241-2 [6] includes the "Privileged User management", which is not relevant for the TOE, because privileged users and corresponding roles are hard-coded in the static TOE configuration file.
The PP 419 241-2 [6] includes the "Change of TOE configuration", which is not relevant for the TOE, because the TOE configuration is a static text file and TOE management functions do not change the configuration.

FAU_GEN.1.2	The TSF shall record within each audit record at least the following information: a) Date and time of the event, type of event, subject identity (if applicable), and the outcome (success or failure) of the event; and b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST: *1)* type of action performed (success or failure), 2) identity of the role which performs the operation. [a]

[a] assignment: other audit relevant information

7.3.2 Cryptographic support (FCS)

7.3.2.1 Cryptographic key generation (FCS_CKM.1)

The FCS_CKM.1 is iterated for different types of generated keys.

7.3.2.1.1 FCS_CKM.1/RSA_SVD – Cryptographic key generation

First of all, TOE generates the D.SVD from the shares of public key (D.clientModulus and D.serverModulus).

FCS_CKM.1.1/RSA_SVD	The TSF shall generate **D.SVD** ~~cryptographic keys~~ in accordance with a specified cryptographic key generation algorithm *TSSP compound public key generation from shares of the public key* [a] and specified cryptographic key sizes *4094, 4095, 4096, 6142, 6143, 6144, 8190, 8191 and 8192 bits* [b] that meet the following: *standard RFC8017 [20] (section 3.1) and article [5]* [c]

[a] assignment: cryptographic key generation algorithm [b] assignment: cryptographic key sizes [c] assignment: list of standards

7.3.2.1.2 FCS_CKM.1/RSA_KTK – Cryptographic key generation

The D.KTK is RSA key pair, which is used to authenticate TOE to the TSE, when initiating the secure channel between the TSE and TOE. TOE uses the HSM to generate and protect the key pair, therefore the reference to the FIPS 140-2 [14] compliant HSM has been included.

FCS_CKM.1.1/RSA_KTK	The TSF shall generate **RSA key pair D.KTK** ~~cryptographic keys in accordance~~ with a ~~specified cryptographic key generation algorithm~~ *FIPS 140-2 [14] compliant HSM*[a] and specified cryptographic key sizes *2048 bits up to 16384 bits*[b] that meet the following: *standard RFC8017 [20]*[c]

[a] assignment: cryptographic key generation algorithm [b] assignment: cryptographic key sizes [c] assignment: list of standards

> Application Note 7
>
> The TOE is expected to use a HSM certified in conformance with FIPS 140-2 [14], see also OE.CSPRNG, OE.HSM.SCD_Unique and OE.HSM.Sig_Secure for key generation. Although the TSF may not generate keys itself, this SFR expresses the requirement for the TSF to invoke the HSM with the appropriate parameters whenever key generation is required.

7.3.2.1.3 FCS_CKM.1/DH_TEK – Cryptographic key generation

The D.TEK is symmetric encryption/decryption and integrity protection key, which is used to create the secure communication channel between the TSE and TOE. D.TEK is generated with a variant of Diffie-Hellman key agreement protocols:

FCS_CKM.1.1/DH_TEK	The TSF shall generate **D.TEK** ~~cryptographic keys~~ with a specified cryptographic key generation algorithm *Diffie-Hellman station-to-station protocol and concatKDF*[a] and specified cryptographic key sizes *2048 bits up to 16384 bits*[b] that meet the following: *standards RFC2631 [21], RFC3526 [22] and SP 800-56A Rev. 2 [23] (section 5.8.1)*[c].

[a] assignment: cryptographic key generation algorithm [b] assignment: cryptographic key sizes [c] assignment: list of standards

7.3.2.1.4 FCS_CKM.1/AES_KWK – Cryptographic key generation

The D.KWK is symmetric encryption/decryption and integrity protection key, which is used to wrap the key material in the TOE database. TOE uses the HSM to generate and protect the key, therefore the reference to the FIPS 140-2 [14] compliant HSM has been included.

FCS_CKM.1.1/AES_KWK	The TSF shall generate **D.KWK** ~~cryptographic keys in accordance~~ with a ~~specified cryptographic key generation algorithm~~ *FIPS 140-2 [14] compliant HSM*[a] and specified cryptographic key sizes *128 bits*[b] that meet the following: *standard SP 800-133 [24]*[c]

[a] assignment: cryptographic key generation algorithm [b] assignment: cryptographic key sizes [c] assignment: list of standards

Application Note 8

The TOE is expected to use a HSM certified in conformance with FIPS 140-2 [14], see also OE.CSPRNG, OE.HSM.SCD_Unique and OE.HSM.Sig_Secure for key generation. Although the TSF may not generate keys itself, this SFR expresses the requirement for the TSF to invoke the HSM with the appropriate parameters whenever key generation is required.

7.3.2.1.5 FCS_CKM.1/AES_DEK – Cryptographic key generation

The D.DEK is symmetric encryption/decryption and integrity protection key, which is used to wrap the sensitive and confidential attributes in the TOE database. TOE generates the D.DEK by himself, but uses the D.KWK to wrap the key for storage.

FCS_CKM.1.1/AES_DEK	The TSF shall generate **D.DEK** ~~cryptographic keys~~ in accordance with a specified cryptographic key generation algorithm *SP 800-133 [24] (section 5)*[a] and specified cryptographic key sizes *128 bits*[b] that meet the following: *standard SP 800-133 [24]*[c]

[a] assignment: cryptographic key generation algorithm [b] assignment: cryptographic key sizes [c] assignment: list of standards

7.3.2.2 Cryptographic key destruction (FCS_CKM.4)

TOE uses same key destruction method for all kind of keys, regardless whether they are stored only in the memory of TOE, in the database or they are encrypted by the HSM:

FCS_CKM.4.1	The TSF shall destroy cryptographic keys in accordance with a ~~specified~~ cryptographic key destruction method *zeroization*[a] that meets the following: *standard FIPS 140-2 [14]*[b].

[a] assignment: cryptographic key destruction method [b] assignment: list of standards

Application Note 9

Note that this is only preliminary destroying and only meant for the operational data. The encrypted key blobs in the database backups are still retained.

7.3.2.3 Cryptographic operation (FCS_COP.1)

The FCS_COP.1 is iterated for different type of cryptographic operations. TOE uses cryptography in multiple areas as follows.

7.3.2.3.1 FCS_COP.1/RSA_SCD – Cryptographic operation

The RSA signature generation and verification algorithm is used in two cases. To generate the compound signature of the Signer (D.signature) TOE uses the RSA signature computation algorithm as defined in TSSP description:

FCS_COP.1.1/RSA_SCD	The TSF shall perform *RSA signature generation*[a] in accordance with a specified cryptographic algorithm *TSSP compound signature generation from the signature shares*[b] and cryptographic key sizes *2047, 2048, 3071, 3072, 4095 and 4096 bits*[c] that meet the following: *standard RFC8017 [20] (method RSASSA-PKCS1-v1_5) and article [5]*[d].

[a] assignment: list of cryptographic operations [b] assignment: cryptographic algorithm [c] assignment: cryptographic key sizes [d] assignment: list of standards

7.3.2.3.2 FCS_COP.1/RSA_Other – Cryptographic operation

In addition to Signer's signatures, TOE also uses RSA algorithm to perform message decryption and encryption and generation and verification of signatures, when securing the communication between TOE and Signer. TOE uses the algorithms in RFC8017 [20] for that.

FCS_COP.1.1/RSA_Other	The TSF shall perform *RSA decryption, encryption, signature generation and verification*[a] in accordance with a specified cryptographic algorithm *RSASSA-PKCS1-v1_5 and RSAES-OAEP*[b] and cryptographic key sizes *2048 bits up to 16384 bits*[c] that meet the following: *standard RFC8017 [20] (method RSASSA-PKCS1-v1_5 and RSAES-OAEP)*[d].

[a] assignment: list of cryptographic operations [b] assignment: cryptographic algorithm [c] assignment: cryptographic key sizes [d] assignment: list of standards

7.3.2.3.3 FCS_COP.1/AES – Cryptographic operation

Encryption and decryption is performed with AES algorithm:

FCS_COP.1.1/AES	The TSF shall perform *encryption and decryption*[a] in accordance with a specified cryptographic algorithm *AES*[b] and cryptographic key sizes *128 bits*[c] that meet the following: *standard FIPS 197 [25]*[d].

[a] assignment: list of cryptographic operations [b] assignment: cryptographic algorithm [c] assignment: cryptographic key sizes [d] assignment: list of standards

7.3.2.3.4 FCS_COP.1/HMAC – Cryptographic operation

Integrity protection and verification is performed with keyed HMAC algorithm:

FCS_COP.1.1/HMAC	The TSF shall perform *integrity protection and verification*[a] in accordance with a specified cryptographic algorithm *HMAC*[b] and cryptographic key sizes *128 bits*[c] that meet the following: *standard FIPS 198-1 [26]*[d].

[a] assignment: list of cryptographic operations [b] assignment: cryptographic algorithm [c] assignment: cryptographic key sizes [d] assignment: list of standards

7.3.2.3.5 FCS_COP.1/SHA-2 – Cryptographic operation

Digest computation is performed with SHA-2 family of algorithm (SHA-256):

FCS_COP.1.1/SHA-2	The TSF shall perform *digest computation*[a] in accordance with a specified cryptographic algorithm *SHA-2*[b] and cryptographic key sizes *256 bits*[c] that meet the following: *standard FIPS 180-4 [27]*[d].

[a] assignment: list of cryptographic operations [b] assignment: cryptographic algorithm [c] assignment: cryptographic key sizes [d] assignment: list of standards

7.3.3 User data protection (FDP)

7.3.3.1 Access control policy and rules (FDP_ACC.1 and FDP_ACF.1)

7.3.3.1.1 FDP_ACC.1/Signer – Subset access control

FDP_ACC.1.1/Signer	The TSF shall enforce the *SFP/Signer*[a] on *list of subjects, objects and operations as specified in the table 16 in section 7.2.3 - SFP/Signer*[b].

[a] assignment: access control SFP [b] assignment: list of subjects, objects, and operations among subjects and objects covered by the SFP

7.3.3.1.2 FDP_ACF.1/Signer – Security attribute based access control

FDP_ACF.1.1/Signer	The TSF shall enforce the *SFP/Signer*[a] to objects based on the following: *list of rules as specified in the table 16 in section 7.2.3 - SFP/Signer*[b].

[a] assignment: access control SFP [b] assignment: list of subjects and objects controlled under the indicated SFP, and for each, the SFP-relevant security attributes, or named groups of SFP-relevant security attributes

FDP_ACF.1.2/Signer	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: *list of rules as specified in the table 16 in section 7.2.3 - SFP/Signer*[a].

[a] assignment: rules governing access among controlled subjects and controlled objects using controlled operations on controlled objects

FDP_ACF.1.3/Signer	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: *none*[a].

[a] assignment: rules, based on security attributes, that explicitly authorise access of subjects to objects

FDP_ACF.1.4/Signer	The TSF shall explicitly deny access of subjects to objects based on the following additional rules: *none*[a].

[a] assignment: rules, based on security attributes, that explicitly authorise access of subjects to objects

7.3.3.1.3 FDP_ACC.1/App – Subset access control

FDP_ACC.1.1/App	The TSF shall enforce the *SFP/App*[a] on *list of subjects, objects and operations as specified in the table 18 in section 7.2.4 - SFP/App*[b].

[a] assignment: access control SFP objects covered by the SFP [b] assignment: list of subjects, objects, and operations among subjects and

7.3.3.1.4 FDP_ACF.1/App – Security attribute based access control

FDP_ACF.1.1/App	The TSF shall enforce the *SFP/App*[a] to objects based on the following: *list of rules as specified in the table 18 in section 7.2.4 - SFP/App*[b].

[a] assignment: access control SFP [b] assignment: list of subjects and objects controlled under the indicated SFP, and for each, the SFP-relevant security attributes, or named groups of SFP-relevant security attributes

FDP_ACF.1.2/App	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: *list of rules as specified in the table 18 in section 7.2.4 - SFP/App*[a].

[a] assignment: rules governing access among controlled subjects and controlled objects using controlled operations on controlled objects

FDP_ACF.1.3/App	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: *none*[a].

[a] assignment: rules, based on security attributes, that explicitly authorise access of subjects to objects

FDP_ACF.1.4/App	The TSF shall explicitly deny access of subjects to objects based on the following additional rules: *none*[a].

[a] assignment: rules, based on security attributes, that explicitly authorise access of subjects to objects

7.3.3.1.5 FDP_ACC.1/Anonymous – Subset access control

FDP_ACC.1.1/Anonymous	The TSF shall enforce the *SFP/Anonymous*[a] on *list of subjects, objects and operations as specified in the table 20 in section 7.2.5 - SFP/Anonymous*[b].

[a] assignment: access control SFP objects covered by the SFP [b] assignment: list of subjects, objects, and operations among subjects and

7.3.3.1.6 FDP_ACF.1/Anonymous – Security attribute based access control

FDP_ACF.1.1/Anonymous	The TSF shall enforce the *SFP/Anonymous*[a] to objects based on the following: *list of rules as specified in the table 20 in section 7.2.5 - SFP/Anonymous*[b].

[a] assignment: access control SFP [b] assignment: list of subjects and objects controlled under the indicated SFP, and for each, the SFP-relevant security attributes, or named groups of SFP-relevant security attributes

FDP_ACF.1.2/Anonymous	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: *list of rules as specified in the table 20 in section 7.2.5 - SFP/Anonymous*[a].

[a] assignment: rules governing access among controlled subjects and controlled objects using controlled operations on controlled objects

FDP_ACF.1.3/Anonymous	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: *none*[a].

[a] assignment: rules, based on security attributes, that explicitly authorise access of subjects to objects

FDP_ACF.1.4/Anonymous	The TSF shall explicitly deny access of subjects to objects based on the following additional rules: *none*[a].

[a] assignment: rules, based on security attributes, that explicitly authorise access of subjects to objects

7.3.3.1.7 FDP_ACC.1/Admin – Subset access control

FDP_ACC.1.1/Admin	The TSF shall enforce the *SFP/Admin*[a] on *list of subjects, objects and operations as specified in the table 21 in section 7.2.6 - SFP/Admin*[b].

[a] assignment: access control SFP objects covered by the SFP [b] assignment: list of subjects, objects, and operations among subjects and

7.3.3.1.8 FDP_ACF.1/Admin – Security attribute based access control

FDP_ACF.1.1/Admin	The TSF shall enforce the *SFP/Admin*[a] to objects based on the following: *list of rules as specified in the table 21 in section 7.2.6 - SFP/Admin*[b].

[a] assignment: access control SFP [b] assignment: list of subjects and objects controlled under the indicated SFP, and for each, the SFP-relevant security attributes, or named groups of SFP-relevant security attributes

FDP_ACF.1.2/Admin	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: *list of rules as specified in the table 21 in section 7.2.6 - SFP/Admin*[a].

[a] assignment: rules governing access among controlled subjects and controlled objects using controlled operations on controlled objects

FDP_ACF.1.3/Admin	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: *none*[a].

[a] assignment: rules, based on security attributes, that explicitly authorise access of subjects to objects

FDP_ACF.1.4/Admin	The TSF shall explicitly deny access of subjects to objects based on the following additional rules: *none*[a].

[a] assignment: rules, based on security attributes, that explicitly authorise access of subjects to objects

7.3.3.1.9 FDP_ACC.1/CA – Subset access control

FDP_ACC.1.1/CA	The TSF shall enforce the *SFP/CA*[a] on *list of subjects, objects and operations as specified in the table 23 in section 7.2.7 - SFP/CA*[b].

[a] assignment: access control SFP [b] assignment: list of subjects, objects, and operations among subjects and objects covered by the SFP

7.3.3.1.10 FDP_ACF.1/CA – Security attribute based access control

FDP_ACF.1.1/CA	The TSF shall enforce the *SFP/CA*[a] to objects based on the following: *list of rules as specified in the table 23 in section 7.2.7 - SFP/CA*[b].

[a] assignment: access control SFP [b] assignment: list of subjects and objects controlled under the indicated SFP, and for each, the SFP-relevant security attributes, or named groups of SFP-relevant security attributes

FDP_ACF.1.2/CA	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: *list of rules as specified in the table 23 in section 7.2.7 - SFP/CA*[a].

[a] assignment: rules governing access among controlled subjects and controlled objects using controlled operations on controlled objects

FDP_ACF.1.3/CA	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: *none*[a].

[a] assignment: rules, based on security attributes, that explicitly authorise access of subjects to objects

FDP_ACF.1.4/CA	The TSF shall explicitly deny access of subjects to objects based on the following additional rules: *none*[a].

[a] assignment: rules, based on security attributes, that explicitly authorise access of subjects to objects

7.3.4 Identification and authentication (FIA)

7.3.4.1 Authentication failure handling (FIA_AFL)

7.3.4.1.1 FIA_AFL.1 – Authentication failure handling

Authentication failure handling is defined for the following authentication events:

FIA_AFL.1.1	The TSF shall detect when *a) 3, b) 6, or c) 9*[a] unsuccessful authentication attempts occur related to *Signer authentication with knowledge-based authentication factor*[b].

[a] selection: [assignment: positive integer number, an administrator configurable positive integer within[assignment: range of acceptable values] [b] assignment: list of authentication events

FIA_AFL.1.2	When the defined number of unsuccessful authentication attempts has been *surpassed*[a], the TSF shall *lock the user account for a) 3 hours, b) 24 hours, or c) disable the user account*[b].

[a] selection: met, surpassed [b] assignment: list of actions

7.3.4.2 Timing of identification and authentication (FIA_UID.1 and FIA_UAU.1)

Some TOE functions can be accessed without identification and authentication, as shown in the following sections:

7.3.4.2.1 FIA_UID.1 – Timing of identification

FIA_UID.1.1	The TSF shall allow *operations 'initiateKey', 'getKeyState', 'getFreshnessToken', 'revokeKey'*[a] on behalf of the user to be performed before the user is identified.

[a] assignment: list of TSF-mediated actions

FIA_UID.1.2	The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

7.3.4.2.2 FIA_UAU.1 – Timing of authentication

FIA_UAU.1.1	The TSF shall allow *operations 'initiateKey', 'getKeyState', 'getFreshnessToken', 'revokeKey'*[a] on behalf of the user to be performed before the user is authenticated.

[a] assignment: list of TSF-mediated actions

FIA_UAU.1.2	The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

> Application Note 10
>
> FIA_UAU.1 requires all users to be authenticated including U.Admin and U.CA as well.

7.3.4.3 Multifactor unforgeable authentication (FIA_UAU.3 and FIA_UAU.4)

7.3.4.3.1 FIA_UAU.3 - Unforgeable authentication

FIA_UAU.3.1	The TSF shall *prevent*[a] use of authentication data that has been forged by any user of the TSF.

[a] selection: detect, prevent

FIA_UAU.3.2	The TSF shall *detect*[a] use of authentication data that has been copied from any other user of the TSF.

[a] selection: detect, prevent

> Application Note 11
>
> Note that the SFR FIA_UAU.3 has been traditionally used with biometric authentication in the context, where the TSF shall be able to detect the forged biometric data. In our case, the TSF is able to prevent the forged digital signatures and copied one-time-passwords.

7.3.4.3.2 FIA_UAU.4/Signer - Single-use authentication mechanisms

FIA_UAU.4.1/Signer	The TSF shall prevent reuse of authentication data related to *Signer authentication*[a].

[a] assignment: identified authentication mechanism(s)

7.3.4.3.3 FIA_UAU.4/App - Single-use authentication mechanisms

FIA_UAU.4.1/App	The TSF shall prevent reuse of authentication data related to *App authentication*[a].

[a] assignment: identified authentication mechanism(s)

> Application Note 12
>
> The authentication methods, which are used to authenticate Signers and Apps, use one-time-passwords and the TSF can prevent the re-use of the old passwords.

7.3.4.3.4 FIA_UAU.5/Signer - Multiple authentication mechanisms

FIA_UAU.5.1/Signer	The TSF shall provide *knowledge-based and possession-based authentication mechanism*[a] to support **U.User** ~~user~~ authentication.

[a] assignment: list of multiple authentication mechanisms

FIA_UAU.5.2/Signer	The TSF shall authenticate **U.User's** ~~any user's~~ claimed identity according to the *following input information and algorithm:*[a] 1. **claimed D.Signing_Key_Id value (to identify the user)** 2. **transmitted D.OTP (possession-based factor)** 3. **JWE envelope encrypted with correct D.TEK (possession-based factor)** 4. **transmitted D.applicationSignaturePart computed on D.DTBS/R with D.clientPart, decrypted with the correct D.PIN (knowledge-based factor)** **authentication algorithm works as follows:** 1. **TOE receives the operation performSignature() request and parses the JWE envelope.** 2. **TOE takes the claimed D.Signing_Key_Id value from the JWE header and retrieves the D.OTP and D.TEK of the corresponding D.Signing_Key_Id object from the database.** 3. **TOE verifies that the JWE envelope is encrypted with the same D.TEK and decrypts the envelope contents.** 4. **TOE verifies that the D.OTP inside the envelope and the D.OTP from the database match.** 5. **TOE retrieves the D.serverPart and D.clientModulus of the corresponding D.Signing_Key_Id from the database, computes the D.serverSignaturePart on the presented D.DTBS/R. TOE then combines D.applicationSignaturePart and D.serverSignature Part to the D.applicationSignatureShare and verifies it with D.clientModulus.** **In case the steps 3, 4 and 5 give positive match, the authentication result is positive, TOE binds U.User with subject S.Signer and role R.Signer. S.Signer is identified with the value of D.Signing_Key_Id.**

[a] assignment: rules describing how the multiple authentication mechanisms provide authentication

7.3.4.3.5 FIA_UAU.5/App - Multiple authentication mechanisms

FIA_UAU.5.1/App	The TSF shall provide *possession-based encryption key and one-time-password authentication mechanism*[a] to support **U.User** ~~user~~ authentication.

[a] assignment: list of multiple authentication mechanisms

FIA_UAU.5.2/App	The TSF shall authenticate **U.User's** ~~any user's~~ claimed identity according to the *following input information and algorithm:*[a] 1. **claimed D.Signing_Key_Id value** 2. **transmitted D.OTP (possession-based factor)** 3. **JWE envelope encrypted with correct D.TEK (possession-based factor)** **authentication algorithm works as follows:** 1. **TOE receives the operation performSignature() request and parses the JWE envelope.** 2. **TOE takes the claimed D.Signing_Key_Id value from the JWE header and retrieves the D.OTP and D.TEK of the corresponding D.Signing_Key_Id object from the database.** 3. **TOE verifies that the JWE envelope is encrypted with the same D.TEK and decrypts the envelope contents.** 4. **TOE verifies that the D.OTP inside the envelope and the D.OTP from the database match.** **In case the steps 3 and 4 give positive match, the authentication result is positive, TOE binds U.User with subject S.App and role R.App. S.App is identified with the value of D.Signing_Key_Id.**

[a] assignment: rules describing how the multiple authentication mechanisms provide authentication

7.3.5 Security Management (FMT)

7.3.5.1 Management of security attributes (FMT_MSA)

7.3.5.1.1 FMT_MSA.1/Signer – Management of security attributes

FMT_MSA.1.1/Signer	The TSF shall enforce the *SFP/Signer*[a] to restrict the ability to *query*[b] the security attributes *listed in the section 7.2.3 – SFP/Signer , in Table 17*[c] to *role R.Signer*[d].

[a] assignment: access control SFP(s), information flow control SFP(s) [b] selection: change_default, query, modify, delete, [assignment: other operations] [c] assignment: list of security attributes [d] assignment: the authorised identified roles

7.3.5.1.2 FMT_MSA.1/App – Management of security attributes

FMT_MSA.1.1/App	The TSF shall enforce the *SFP/App*[a] to restrict the ability to *query, modify, delete*[b] the security attributes *listed in the section 7.2.4 – SFP/App , in Table 19*[c] to *role R.App*[d].

[a] assignment: access control SFP(s), information flow control SFP(s) [b] selection: change_default, query, modify, delete, [assignment: other operations] [c] assignment: list of security attributes [d] assignment: the authorised identified roles

7.3.5.1.3 FMT_MSA.1/Admin – Management of security attributes

FMT_MSA.1.1/Admin	The TSF shall enforce the *SFP/Admin*[a] to restrict the ability to *modify*[b] the security attributes *listed in the section 7.2.6 – SFP/Admin , in Table 22*[c] to *role R.Admin*[d].

[a] assignment: access control SFP(s), information flow control SFP(s) [b] selection: change_default, query, modify, delete, [assignment: other operations] [c] assignment: list of security attributes [d] assignment: the authorised identified roles

7.3.5.1.4 FMT_MSA.1/CA – Management of security attributes

FMT_MSA.1.1/CA	The TSF shall enforce the *SFP/CA*[a] to restrict the ability to *query, delete*[b] the security attributes *listed in the section 7.2.7 – SFP/CA , in Table 24*[c] to *role R.CA*[d].

[a] assignment: access control SFP(s), information flow control SFP(s) [b] selection: change_default, query, modify, delete, [assignment: other operations] [c] assignment: list of security attributes [d] assignment: the authorised identified roles

7.3.5.1.5 FMT_MSA.2 – Secure security attributes

FMT_MSA.2.1	The TSF shall ensure that only secure values are accepted for *attributes listed in the section 7.1.2 – TSF data*[a].

[a] assignment: list of security attributes

7.3.5.1.6 FMT_MSA.3 – Static attribute initialisation

FMT_MSA.3.1	The TSF shall enforce the *SFP/Init*[a] to provide *restrictive*[b] default values for security attributes that are used to enforce the SFP.

[a] assignment: access control SFP(s), information flow control SFP(s) [b] selection, choose one of: restrictive, permissive, [assignment: other property]

FMT_MSA.3.2	The TSF shall allow *no role*[a] to specify alternative initial values to override the default values when an object or information is created.

[a] assignment: the authorised identified roles

7.3.5.2 Management of TSF data (FMT_MTD)

7.3.5.2.1 FMT_MTD.1 – Management of TSF data

FMT_MTD.1.1	The TSF shall restrict the ability to *change_default, query, modify, delete*[a] the *attributes listed in the section 7.1.2 – TSF data*[b] to *R.Signer, R.App, R.Admin, R.CA, R.Anonymous*[c]

[a] selection: change_default, query, modify, delete, [assignment: other operations] [b] assignment: list of TSF data [c] assignment: the authorised identified roles

7.3.5.3 Specification of management functions (FMT_SMF)

7.3.5.3.1 FMT_SMF.1 – Specification of Management Functions

FMT_SMF.1.1	The TSF shall be capable of performing the following management functions: *listed operations in the section 7.2.1 – Operations*[a].

[a] assignment: list of management functions to be provided by the TSF

7.3.5.4 Security management roles (FMT_SMR)

7.3.5.4.1 FMT_SMR.1 – Security roles

FMT_SMR.1.1	The TSF shall maintain the roles *R.Signer, R.App, R.Admin, R.CA, R.Anonymous*[a].

[a] assignment: the authorised identified roles

FMT_SMR.1.2	The TSF shall be able to associate users with roles.

7.3.6 Protection of the TSF (FPT)

7.3.6.1 Confidentiality and integrity of transmitted TSF data (FPT_ITC and FPT_ITI)

7.3.6.1.1 FPT_ITC.1 – Inter-TSF confidentiality during transmission

FPT_ITC.1.1	The TSF shall protect all TSF data transmitted from the TSF to another trusted IT product from unauthorised disclosure during transmission.

7.3.6.1.2 FPT_ITI.1 – Inter-TSF detection of modification

FPT_ITI.1.1	The TSF shall provide the capability to detect modification of all TSF data during transmission between the TSF and another trusted IT product within the following metric: *HMAC integrity protection*[a].

[a] assignment: a defined modification metric

FPT_ITI.1.2	The TSF shall provide the capability to verify the integrity of all TSF data transmitted between the TSF and another trusted IT product and perform *operation abortion*[a] if modifications are detected.

[a] assignment: action to be taken

7.3.6.2 Replay detection (FPT_RPL)

7.3.6.2.1 FPT_RPL.1 – Replay detection

FPT_RPL.1.1	The TSF shall detect replay for the following entities: *Signer*[a].

[a] assignment: list of identified entities

FPT_RPL.1.2	The TSF shall perform *key pair destroying*[a] when replay is detected.

[a] assignment: list of specific actions

7.3.7 Trusted path (FTP)

7.3.7.1 Confidentiality and integrity of transmitted TSF data (FTP_ITC)

7.3.7.1.1 FTP_ITC.1 – Inter-TSF trusted channel

FTP_ITC.1.1	The TSF shall provide a communication channel between itself and another trusted IT product 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.

FTP_ITC.1.2	The TSF shall permit *the TSF*[a] to initiate communication via the trusted channel.

[a] selection: the TSF, another trusted IT product

FTP_ITC.1.3	The TSF shall initiate communication via the trusted channel for *operations with database and operations with HSM*[a].

[a] assignment: list of functions for which a trusted channel is required

7.3.7.2 Confidentiality and integrity of communication with users (FTP_TRP)

7.3.7.2.1 FTP_TRP.1 – Trusted path

FTP_TRP.1.1	The TSF shall provide a communication path between itself and **Signer** *remote*[a] users that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from *modification, disclosure, replay attack*[b].

[a] selection: remote, local [b] selection: modification, disclosure, [assignment: other types of integrity or confidentiality violation]

FTP_TRP.1.2	The TSF shall permit **Signer** *remote users*[a] to initiate communication via the trusted path.

[a] selection: the TSF, local users, remote users

FTP_TRP.1.3	The TSF shall require the use of the trusted path for *all operations requested by users*[a].

[a] selection: initial user authentication, [assignment: other services for which trusted path is required]

7.4 Security Requirements Rationale

7.4.1 Mapping between SFRs and TOE Security Objectives

The mapping of TOE Security Objectives to SFRs is shown in the table 25.

Table 25. Mapping between TOE security objectives and SFRs

	OT.SCD_Confidential	OT.Sig_Secure	OT.SCD/SVD_Corresp	OT.TSSP_End2End	OT.SAP_Replay_Protection	OT.TSSP_Require_clientSignatureShare	OT.TSSP_Validate_clientSignatureShare	OT.TSSP_CloneDetection	OT.TSSP_TimeDelay_Locks	OT.DTBS/R_Protect	OT.Audit_Events	OT.Privileged_User_Management	OT.Privileged_User_Authentication	OT.Privileged_User_Protection
FAU_GEN.1											X			
FCS_CKM.1/RSA_SVD			X											
FCS_CKM.1/RSA_KTK				X										
FCS_CKM.1/DH_TEK				X										
FCS_CKM.1/AES_KWK	X													
FCS_CKM.1/AES_DEK	X							X	X					
FCS_CKM.4	X			X										
FCS_COP.1/RSA_SCD		X				X								
FCS_COP.1/RSA_Other				X										
FCS_COP.1/AES	X			X	X									
FCS_COP.1/HMAC	X			X	X									
FCS_COP.1/SHA-2	X	X		X	X									

Table 25. Mapping between TOE security objectives and SFRs

	OT.SCD_Confidential	OT.Sig_Secure	OT.SCD/SVD_Corresp	OT.TSSP_End2End	OT.SAP_Replay_Protection	OT.TSSP_Require_clientSignatureShare	OT.TSSP_Validate_clientSignatureShare	OT.TSSP_CloneDetection	OT.TSSP_TimeDelay_Locks	OT.DTBS/R_Protect	OT.Audit_Events	OT.Privileged_User_Management	OT.Privileged_User_Authentication	OT.Privileged_User_Protection
FDP_ACC.1/Signer FDP_ACF.1/Signer						X	X							
FDP_ACC.1/App FDP_ACF.1/App							X	X						
FDP_ACC.1/Anonymous FDP_ACF.1/Anonymous		X												
FDP_ACC.1/Admin FDP_ACF.1/Admin												X		
FDP_ACC.1/CA FDP_ACF.1/CA		X												
FIA_AFL.1									X					
FIA_UID.1 FIA_UAU.1				X	X	X	X	X	X				X	
FIA_UAU.3							X							
FIA_UAU.4/Signer					X			X						
FIA_UAU.4/App					X			X						
FIA_UAU.5/Signer				X		X	X	X	X					
FIA_UAU.5/App								X						
FMT_MSA.1/Signer				X	X			X	X					
FMT_MSA.1/App				X	X			X	X					
FMT_MSA.1/Admin												X		
FMT_MSA.1/CA												X		
FMT_MSA.2	X	X		X	X							X	X	
FMT_MSA.3	X			X					X				X	
FMT_MTD.1												X		X
FMT_SMF.1								X				X		
FMT_SMR.1												X		X
FPT_RPL.1					X			X						
FPT_ITC.1 FPT_ITI.1 FTP_ITC.1	X	X		X	X					X				
FTP_TRP.1				X	X					X				

7.4.2 SFR Rationale

Here below are the rationale about the satisfaction of security objectives for TOE by TOE SFRs.

OT.SCD_Confidential FCS_CKM.1/AES_KWK ensures that all keys stored in the database are protected in integrity. FCS_CKM.1/AES_DEK ensures that all confidential data that is stored in the database is protected in integrity. FCS_CKM.4 ensures that all the keys used for securing the data are destructed in case of zeroisation. FCS_COP.1/AES ensures that encryption and decryption of confidential data is performed with AES algorithm. FCS_COP.1/HMAC ensures that integrity protection and verification is performed with HMAC algorithm. FCS_COP.1/SHA-2 ensures that digest computation is performed with SHA-2 family of algorithms. FMT_MSA.2 and FMT_MSA.3 ensures that only secure values are accepted for security attributes. FPT_ITI.1, FPT_ITC.1 and FTP_ITC.1 ensures integrity and confidentiality protection during transmission of data.

OT.Sig_Secure FCS_COP.1/RSA_SCD ensures that TOE generates the compound signature of the Signer (D.signature). TOE uses the RSA signature computation algorithm FCS_COP.1/SHA-2 ensures that digest computation is performed with SHA-2 family of algorithms. FDP_ACC.1/Anonymous and FDP_ACF.1/Anonymous ensures that the signer can be created. FDP_ACC.1/CA and FDP_ACF.1/CA ensures that CA can revoke key in case when it's needed. FMT_MSA.2 ensures that only secure values are accepted for security attributes. FPT_ITI.1, FPT_ITC.1 and FTP_ITC.1 ensures integrity and confidentiality protection during transmission of data.

OT.SCD/SVD_Corresp FCS_CKM.1/RSA_SVD ensures that TOE generates the D.SVD from the shares of public key (D.clientModulus and D.serverModulus) using algorithm that meets the standard RFC8017 [20] (section 3.1) and [5].

OT.TSSP_End2End FCS_CKM.1/RSA_KTK ensures the authentication of the TOE to the TSE library. FCS_CKM.1/DH_TEK ensures the secure communication channel between the TOE and the TSE library. FCS_CKM.4 ensures that all the keys used for securing the data are destructed in case of zeroisation. FCS_COP.1/RSA_Other ensures TOE produces technical signatures to secure the communication between TOE and Signer FCS_COP.1/AES ensures that encryption and decryption of confidential data is performed with AES algorithm. FCS_COP.1/HMAC ensures that integrity protection and verification is performed with HMAC algorithm. FCS_COP.1/SHA-2 ensures that digest computation is performed with SHA-2 family of algorithms. FIA_UID.1 and FIA_UAU.1 ensure that the user is identified and or authenticated before each operation if needed. FIA_UAU.5/Signer ensures the signer authenticates itself. FMT_MSA.1/Signer ensures the signer to manage the belonging TSF data. FMT_MSA.1/App ensures the app to manage the belonging TSF data. FMT_MSA.2 and FMT_MSA.3 ensures that only secure values are accepted for security attributes. FPT_ITI.1, FPT_ITC.1 and FTP_ITC.1 ensures integrity and confidentiality protection during transmission of data. FTP_TRP.1 ensures that there is a trusted path used for communication.

OT.SAP_Replay_Protection FCS_COP.1/AES ensures that encryption and decryption of confidential data is performed with AES algorithm. FCS_COP.1/HMAC ensures that integrity protection and verification is performed with HMAC algorithm. FCS_COP.1/SHA-2 ensures that digest computation is performed with SHA-2 family of algorithms. FIA_UID.1 and FIA_UAU.1 ensure that the user is identified and or authenticated before each operation if needed. FIA_UAU.4/Signer ensures there is no reuse of the signer authentication data. FIA_UAU.4/App ensures there is no reuse of the app authentication data. FMT_MSA.1/Signer ensures the signer to manage the belonging TSF data. FMT_MSA.1/App ensures the app to manage the

belonging TSF data. FMT_MSA.2 ensures that only secure values are accepted for security attributes. FPT_RPL.1 ensures there is no reply of D.OTP. FPT_ITI.1, FPT_ITC.1 and FTP_ITC.1 ensures integrity and confidentiality protection during transmission of data. FTP_TRP.1 ensures that there is a trusted path used for communication.

OT.TSSP_Require_clientSignatureShare FCS_COP.1/RSA_SCD ensures that TOE generates the compound signature of the Signer (D.signature). FDP_ACC.1/Signer and FDP_ACF.1/Signer FIA_UID.1 and FIA_UAU.1 ensure that the user is identified and or authenticated before each operation if needed. FIA_UAU.5/Signer ensures that the Signer is authenticated.

OT.TSSP_Validate_clientSignatureShare FDP_ACC.1/Signer and FDP_ACF.1/Signer FDP_ACC.1/App and FDP_ACF.1/App ensures the App can upload the D.clientModulus to SZ. FIA_UID.1 and FIA_UAU.1 ensure that the user is identified and or authenticated before each operation if needed. FIA_UAU.3 ensures that the authentication data was not forged. FIA_UAU.5/Signer ensures that the Signer is authenticated.

OT.TSSP_CloneDetection FCS_CKM.1/AES_DEK ensures that all confidential data is stored in the database is protected in integrity. FDP_ACC.1/App and FDP_ACF.1/App ensures the App can upload the D.OTP to SZ. FIA_UAU.4/Signer ensures there is no reuse of the signer authentication data. FIA_UAU.4/App ensures there is no reuse of the app authentication data. FIA_UID.1 and FIA_UAU.1 ensure that the user is identified and or authenticated before each operation if needed. FIA_UAU.5/Signer ensures that the Signer is authenticated. FIA_UAU.5/App ensures that the App is authenticated. FMT_MSA.1/Signer ensures the signer to manage the belonging TSF data. FMT_MSA.1/App ensures the app to manage the belonging TSF data. FMT_SMF.1 ensures that the app can refresh D.OTP. FPT_RPL.1 ensures there is no reply of D.OTP

OT.TSSP_TimeDelay_Locks FCS_CKM.1/AES_DEK ensures that all confidential data is stored in the database is protected in integrity. FIA_AFL.1 ensure the user Lock after defined number of unsuccessful Signer authentication. FIA_UID.1 and FIA_UAU.1 ensure that the user is identified and or authenticated before each operation if needed. FIA_UAU.5/Signer ensures that the Signer is authenticated. FMT_MSA.1/Signer ensures the signer to manage the belonging TSF data. FMT_MSA.1/App ensures the app to manage the belonging TSF data. FMT_MSA.3 ensures that only secure values are accepted for security attributes.

OT.DTSB/R_Protect FPT_ITI.1, FPT_ITC.1 and FTP_ITC.1 ensures the integrity and confidentiality of the DTBSR, when transmitted to and from external IT components. FTP_TRP.1 ensures the confidentiality of data providing trusted communicationt path between the TOE and remote users.

OT.Audit_Events FAU_GEN.1 ensures that the TOE creates audit records about the important system events.

OT.Privileged_User_Management FDP_ACC.1/Admin ensures the access control to U.Admin on objects based on sassociated in SFP/Admin. FDP_ACF.1/Admin ensures the access control to U.Admin on security attribute based objects operation associated in SFP/Admin. FMT_MSA.1/Admin ensures the Admin to manage the belonging TSF data. FMT_MSA.1/CA ensures the CA to manage the belonging TSF data. FMT_MSA.2 ensures that only secure values are accepted for security attributes. FMT_MTD.1 ensures the integrity of the TSF data with restrict the management of TSF data to R.Signer, R.App, R.Admin, R.CA, R.Anonymous. FMT_SMF.1 ensures that the Privileged users app can execute their associated

operations. FMT_SMR.1 ensures the maintenance of Privileged roles and associate the Privileged users with roles.

OT.Privileged_User_Authentication FIA_UID.1 and FIA_UAU.1 ensure that the user is identified and or authenticated before each operation if needed. FIA_UAU.1 ensures authentication mechanism to U.Admin user authentication claiming the user to authenticate himself before any action. FMT_MSA.2 and FMT_MSA.3 ensures that only secure values are accepted for security attributes.

OT.Privileged_User_Protection FMT_MTD.1 ensures the integrity of the TSF data with restrict the management of TSF data to R.Signer, R.App, R.Admin, R.CA, R.Anonymous. FMT_SMR.1 ensures the maintenance of of roles R.Signer, R.App, R.Admin, R.CA, R.Anonymous and associate the users with roles.

7.4.3 SFR Dependencies Analysis

Table 26 shows how the dependencies of the SFRs is fulfilled.

Meaning of the wildcards in the SFR iteration are the followings:

- FCS_CKM.1/* = (FCS_CKM.1/RSA_SVD, FCS_CKM.1/RSA_KTK, FCS_CKM.1/DH_TEK, FCS_CKM.1/AES_KWK, FCS_CKM.1/AES_DEK)
- FCS_COP.1/* = (FCS_COP.1/RSA_SCD, FCS_COP.1/RSA_Other, FCS_COP.1/AES, FCS_COP.1/HMAC, FCS_COP.1/SHA-2)
- FDP_ACC.1/* = (FDP_ACC.1/Signer, FDP_ACC.1/App, FDP_ACC.1/Anonymous, FDP_ACC.1/Admin, FDP_ACC.1/CA)
- FDP_ACF.1/* = (FDP_ACF.1/Signer, FDP_ACF.1/App, FDP_ACF.1/Anonymous, FDP_ACF.1/Admin, FDP_ACF.1/CA)
- FIA_UAU.4/* = (FIA_UAU.4/Signer, FIA_UAU.4/App)
- FIA_UAU.5/* = (FIA_UAU.5/Signer, FIA_UAU.5/App)
- FMT_MSA.1/* = (FMT_MSA.1/Signer, FMT_MSA.1/App, FMT_MSA.1/Admin, FMT_MSA.1/CA)

Table 26. Analysis of fulfillment of SFR dependencies

SFR	Dependecies	Fulfilled by
FAU_GEN.1	FPT_STM.1	(See application note 13)
FCS_CKM.1/*	FCS_CKM.2 or FCS_COP.1	FCS_COP.1/*
	FCS_CKM.4	FCS_CKM.4
FCS_CKM.4	FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1	FCS_CKM.1/*
FCS_COP.1/*	FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1	FCS_CKM.1/*
	FCS_CKM.4	FCS_CKM.4
FDP_ACC.1/*	FDP_ACF.1	FDP_ACF.1/*
FDP_ACF.1/*	FDP_ACC.1/*	FDP_ACC.1/*

Table 26. Analysis of fulfillment of SFR dependencies

SFR	Dependecies	Fulfilled by
	FMT_MSA.3	FMT_MSA.3
FIA_AFL.1	FIA_UAU.1	FIA_UAU.1
FIA_UAU.1	FIA_UID.1	FIA_UID.1
FIA_UAU.3	none	
FIA_UAU.4/*	none	
FIA_UAU.5/*	none	
FIA_UID.1	none	
FMT_MSA.1/*	FDP_ACC.1 or FDP_IFC.1	FDP_ACC.1/*
	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	FMT_SMF.1
FMT_MSA.2	FDP_ACC.1 or FDP_IFC.1	FDP_ACC.1/*
	FMT_MSA.1	FMT_MSA.1/*
	FMT_SMR.1	FMT_SMR.1
FMT_MSA.3	FMT_MSA.1	FMT_MSA.1/*
	FMT_SMR.1	FMT_SMR.1
FMT_MTD.1	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	FMT_SMF.1
FMT_SMF.1	none	
FMT_SMR.1	FIA_UID.1	FIA_UID.1
FPT_ITC.1	none	
FPT_ITI.1	none	
FPT_RPL.1	none	
FTP_ITC.1	none	
FTP_TRP.1	none	

Application Note 13

The FAU_GEN.1 dependency on the FPT_STM.1 is not fulfilled, because the TSF relies on operating system to provide trusted timestamps. The environment objective OE.Trusted_Timestamps is ensurig that the operating system is configured to synchronise the clock to the trusted time source.

7.5 Security Assurance Requirements

7.5.1 Rationale for selecting the SARs

The assurance level for this ST is chosen to be the EAL4 augmented. EAL4 permits a developer to gain maximum assurance from positive security engineering based on good commercial development practices, without the need for highly specialised processes and practices. EAL4 is considered to be the highest level that could be applied to the product without undue expense and complexity. As such, EAL4 is appropriate for the commercial products that require moderate to high security functions. The TOE described in this ST (TOE type QSCD) is just such a product.

The EAL4 is augmented by AVA_VAN.5 (Advanced methodical vulnerability analysis). This is chosen because the TOEs of type QSCD must be highly resistant to the penetration attacks to meet the security objectives of the P.SCD_Confidential, P.Sig_unForgeable, P.SCD_userOnly.

7.5.2 Security assurance components

The assurance components are identified in the table 27.

Table 27. Security Assurance Components used in the ST

Assurance Class	Assurance Components
Security Target (ASE)	ST introduction (ASE_INT.1) Conformance claims (ASE_CCL.1) Security problem definition (ASE_SPD.1) Security objectives (ASE_OBJ.2) Extended components definition (ASE_ECD.1) Derived security requirements (ASE_REQ.2) TOE summary specification (ASE_TSS.1)
Development (ADV)	Security architecture description (ADV_ARC.1) Complete functional specification (ADV_FSP.4) Basic modular design (ADV_TDS.3) Implementation representation of the TSF (ADV_IMP.1)
Guidance documents (AGD)	Operational user guidance (AGD_OPE.1) Preparative measures (AGD_PRE.1)
Life-cycle support (ALC)	Production support, acceptance procedures and automation (ALC_CMC.4) Problem tracking CM coverage (ALC_CMS.4) Delivery procedures (ALC_DEL.1) Identification of security measures (ALC_DVS.1) Developer defined life-cycle model (ALC_LCD.1) Well-defined development tools (ALC_TAT.1)
Tests (ATE)	Functional testing (ATE_FUN.1) Analysis of coverage (ATE_COV.2) Testing: basic design (ATE_DPT.1) Independent testing - sample (ATE_IND.2)

7.5.3 SAR dependencies analysis

The assurance level of this ST is EAL4 augmented by AVA_VAN.5 (advanced methodical vulnerability analysis). The component AVA_VAN.5 has the following dependencies:

- ADV_ARC.1 Architectural design with domain separation and non-bypassability
- ADV_FSP.4 Complete functional specification
- ADV_TDS.3 Basic modular design
- ADV_IMP.1 Implementation representation of the TSF
- AGD_OPE.1 Operational user guidance
- AGD_PRE.1 Preparative procedures
- ATE_DPT.1 Testing: basic design

All of these dependencies are met in the EAL4 assurance package.

8 TOE Summary Specification (ASE_TSS)

This section provides the summary information of the Security Functions of the TOE and describes, how the TOE satisfies all the SFRs described in the section 7.3 – Security Functional Requirements. It is meant as a high-level overview of the TOE. For more detailed information, please refer to the technical architecture documents of the SecureZone [8] and other components of the Smart-ID system [7].

8.1 TOE Security Functions

8.1.1 TOE management and access control

8.1.1.1 SF.Authentication

The TOE authenticates users with the following methods:

1. no personalised identification/authentication – for example, the caller to the monitoring interface are not authenticated by TOE and only general uptime, performance and health information about the TOE is provided over the monitoring interface. Also, the caller to some operations with key pairs are not authenticated by TOE. For example, querying the status of the key pair and destroying the key pair is protected by environment and network mechanisms and not by TOE itself. For details, please refer to the section 7.2.5, where the Security Function Policy SFP/Anonymous is defined.
2. S.App authentication with the possession-based authentication data. Basically, TOE is using the user-name and password authentication and the shared cryptographic key D.TEK to authenticate App. The TOE updates the D.OTP for each upcoming key operation and sends the new password to the App. Because the TOE can detect if App is using the wrong D.OTP, this makes the one-time-password a possession-based authentication factor.
3. S.Signer authentication with the possession-based and the knowledge-based authentication data. TOE adds private key-based authentication factor with the proof-of-posession to the S.App authentication. Because Signer has to enter the correct D.PIN to decrypt the local D.clientPart in order to create the D.applicationSignaturePart, this adds the knowledge-based authentication factor.
4. S.Admin authentication with the HSM OCS password.

Additionally, in case TOE receives three consecutive unsuccessful S.Signer authentication tries, TOE locks the key pair for three hours. If additional three tries are received, TOE locks the key pair for 24 hours. If additional three tries are received, TOE destroys the key pair.

This SF implements the following SFRs:

1. FIA_AFL.1 – Authentication failure handling
2. FIA_UID.1 – Timing of identification

3. FIA_UAU.1 – Timing of authentication
4. FIA_UAU.3 – Unforgeable authentication
5. FIA_UAU.4/Signer and FIA_UAU.4/App – Single-use authentication mechanisms
6. FIA_UAU.5/Signer and FIA_UAU.5/App – Multiple authentication mechanisms
7. FPT_RPL.1 - Replay detection

8.1.1.2 SF.AccessControl

In general, TOE is dividing the users into three main groups:

1. anonymous users,
2. key pair owners (Signers),
3. privileged users (Admins and CA)

Anonymous users are allowed to perform some operations, which do not require authorisation. For example, querying the status of the key pair and destroying the key pair is not authenticated by TOE and no special authorisation is required.

The key pair owners are allowed to perform the key pair operations on their own key pair. In case the Signer is authenticated with possession-based and the knowledge-based authentication data, the TOE allows to complete the signature. In case the Signer is only authenticated with possession-based authentication factors, as is the case when the Smart-ID App is performing technical operations behalf of the Signer and the App doesn't request authorisation and the entry of the D.PIN from the Signer, TOE only allows to perform technical operations. This kind of access control follows naturally from the implemention of the TSSP protocol, which requires that in order to complete the Signer's D.signature, one needs the D.applicationSignaturePart and without that, it is mathematically not possible to create the signature.

The "owning" of a key pair is determined first by verifying that the claimed D.Signing_Key_Id and presented passwords and used cryptographic key D.TEK correspond to the information in the TOE database. Additionally, the "owning" of a key pair is also determined mathematically, as the presented D.applicationSignaturePart needs to match with the D.serverSignaturePart. In case somebody would claim ownership of some other key pair, the signature verification with the D.clientModulus would fail. This sort of access control feature also follows naturally from the implemention of the TSSP protocol.

Privileged users can perform key pair operations on any key pair, however, the list of operations is limited to only specific methods. Privileged users are not allowed to invoke signature completion at all, the API which is offered to them doesn't implement such functions at all.

All those rules are described in more detail within the section 7.2 – Security Function Policies (SFP).

This SF implements the following SFRs:

1. FDP_ACC.1/Signer, FDP_ACC.1/App, FDP_ACC.1/Anonymous, FDP_ACC.1/Admin, FDP_ACC.1/CA – Subset access control
2. FDP_ACF.1/Signer, FDP_ACF.1/App, FDP_ACF.1/Anonymous, FDP_ACF.1/Admin, FDP_ACF.1/CA – Security attribute based access control
3. FMT_MSA.1.1/Signer, FMT_MSA.1/App, FMT_MSA.1/Admin – Management of security attributes
4. FMT_MSA.2 – Secure security attributes
5. FMT_MSA.3 – Static attribute initialisation

6. FMT_MTD.1 – Management of TSF data
7. FMT_SMF.1 – Specification of Management Functions
8. FMT_SMR.1 – Security roles

8.1.1.3 SF.Audit – Security audit generation

TOE uses standard Java toolset to generate audit records of the important system events. The audit log is exported to external system.

This SF implements the FAU_GEN.1 – Security audit generation.

8.1.2 Handling of cryptographic material and algorithms

8.1.2.1 SF.KeyGen – Key generation

TOE uses the FIPS 140-2 [14]-certified HSM to perform most of the key generation operations. In case the HSM doesn't support generation and management of particular key type, TOE is generating that by himself.

1. D.SVD – TOE implements the TSSP [5] and generates the compund modulus of the key pair, using modulus multiplcation of D.clientModulus and D.serverModulus
2. D.KTK – TOE uses the HSM to generate the regular RSA key pair. The private key will be encrypted by HSM.
3. D.TEK – TOE implements the Diffie-Hellman key agreement protocol defined in RFC2631 [21] and key derivation function defined in SP 800-56A Rev. 2 [23] (section 5.8.1).
4. D.KWK – TOE uses the HSM to generate the regular AES key. The key will be encrypted by HSM.
5. D.DEK – TOE uses the 3rd party library Bouncy Castle to generate the regular AES key. The key will be wrapped with D.KWK.

This SF implements the following SFRs:

1. FCS_CKM.1/RSA_SVD, FCS_CKM.1/RSA_KTK, FCS_CKM.1/DH_TEK, FCS_CKM.1/AES_KWK, FCS_CKM.1/AES_DEK – Cryptographic key generation

8.1.2.2 SF.CryptoAlgorithms – Using standard cryptographic algorithms

TOE uses the FIPS 140-2 [14]-certified HSM to perform most of the key usage operations. In case the HSM doesn't support operations with the particular key type, TOE is implementing this by himself.

1. computation of the signatures – TOE implements the TSSP [5] and generates the compound signatures (D.signature) from the shares of signature.
2. creation and verification of RSA signatures – TOE uses the HSM to generate the RSA signature and 3rd party library Bouncy Castle to verify the signatures.
3. encryption/decryption of JWE messages for transmission and database storage – TOE uses the 3rd party library Nimbus to create and verify the JWE messages. The encryption/decryption operations are delegated to the HSM.

This SF implements the following SFRs:

1. FCS_COP.1/RSA_SCD, FCS_COP.1/RSA_Other, FCS_COP.1/AES, FCS_COP.1/HMAC, FCS_COP.1/SHA-2 – Cryptographic operation

8.1.2.3 SF.KeyZer – Key destruction

The TOE destroys the following cryptographic keys after they are no longer used:

1. D.serverPart
2. D.serverShare
3. D.DEK
4. D.TEK
5. D.KWK
6. D.KTK

TOE destroys the keys by overwriting the content of the key storage blob object in the database.

This SF implements the FCS_CKM.4 – Cryptographic key destruction.

8.1.3 Protecting communication with external components

8.1.3.1 SF.TrustedPath – Trusted path with the user

TOE uses JWE messages for communicating with the Smart-ID App TSE. JWE messages are encrypted with the D.TEK and they are integrity protected.

This SF implements the FTP_TRP.1 – Trusted path.

8.1.3.2 SF.SecureChannel – Secure channel with external components

TOE uses vendor-specific proprietary communication channel when connecting with HSM or database, such as nCipher impath and PostgreSQL connections. Those methods provide the cryptographic checksum validation of the integrity for the transmitted data. When TOE detects the modifications and integrity errors with the transmitted data, it aborts the operation.

This SF implements the following SFRs:

1. FTP_ITC.1 – Inter-TSF trusted channel
2. FPT_ITI.1 – Inter-TSF detection of modification
3. FPT_ITC.1 – Inter-TSF confidentiality during transmission

8.2 TOE Summary Specification Rationale

The table 28 shows the mapping between SFRs and TOE Security Functions to provide the quick overview.

Table 28. Mapping between SFRs and TSF

SFR	SF
FAU_GEN.1	SF.Audit
FCS_CKM.1/*	SF.KeyGen
FCS_CKM.4	SF.KeyZer
FCS_COP.1/*	SF.CryptoAlgorithms
FDP_ACC.1/* FDP_ACF.1/*	SF.AccessControl

Table 28. Mapping between SFRs and TSF

SFR	SF
FIA_AFL.1 FIA_UID.1 FIA_UAU.1 FIA_UAU.3 FIA_UAU.4/* FIA_UAU.5/*	SF.Authentication
FIA_MSA.1/* FMT_MSA.2 FMT_MSA.3 FMT_MTD.1 FMT_SMF.1 FMT_SMR.1	SF.AccessControl
FPT_RPL.1	SF.Authentication
FPT_ITC.1 FPT_ITI.1 FTP_ITC.1	SF.SecureChannel
FTP_TRP.1	SF.TrustedPath