CERTIFICATION REPORT

Certification file:	**TUVIT-TSZ-CC-9264-2019**
Product / System:	software component (library) Smart-ID App Threshold Signature Engine, Version 10.3.3
Product manufacturer:	SK ID Solutions AS Pärnu avenue 141 11314 Tallinn, Estonia
Customer:	see above
Evaluation body:	TÜV Informationstechnik GmbH TÜV NORD GROUP Evaluation Body for IT Security Langemarckstr. 20 45141 Essen Germany
Evaluation report:	*Version 2 as of 2019-05-16* project-number: 8114700517 authors: Arzu Sarial, Alexander Bobel
Result:	EAL2
Evaluation stipulations:	none
Certifier:	Dr. Silke Keller
Certification stipulations:	none
Version / Date:	Version 1.0, 2019-05-16

……..……..……	……..……..……
Dr. Christoph Sutter	Dr. Silke Keller
Head of Certification Body	Certifier

Contents

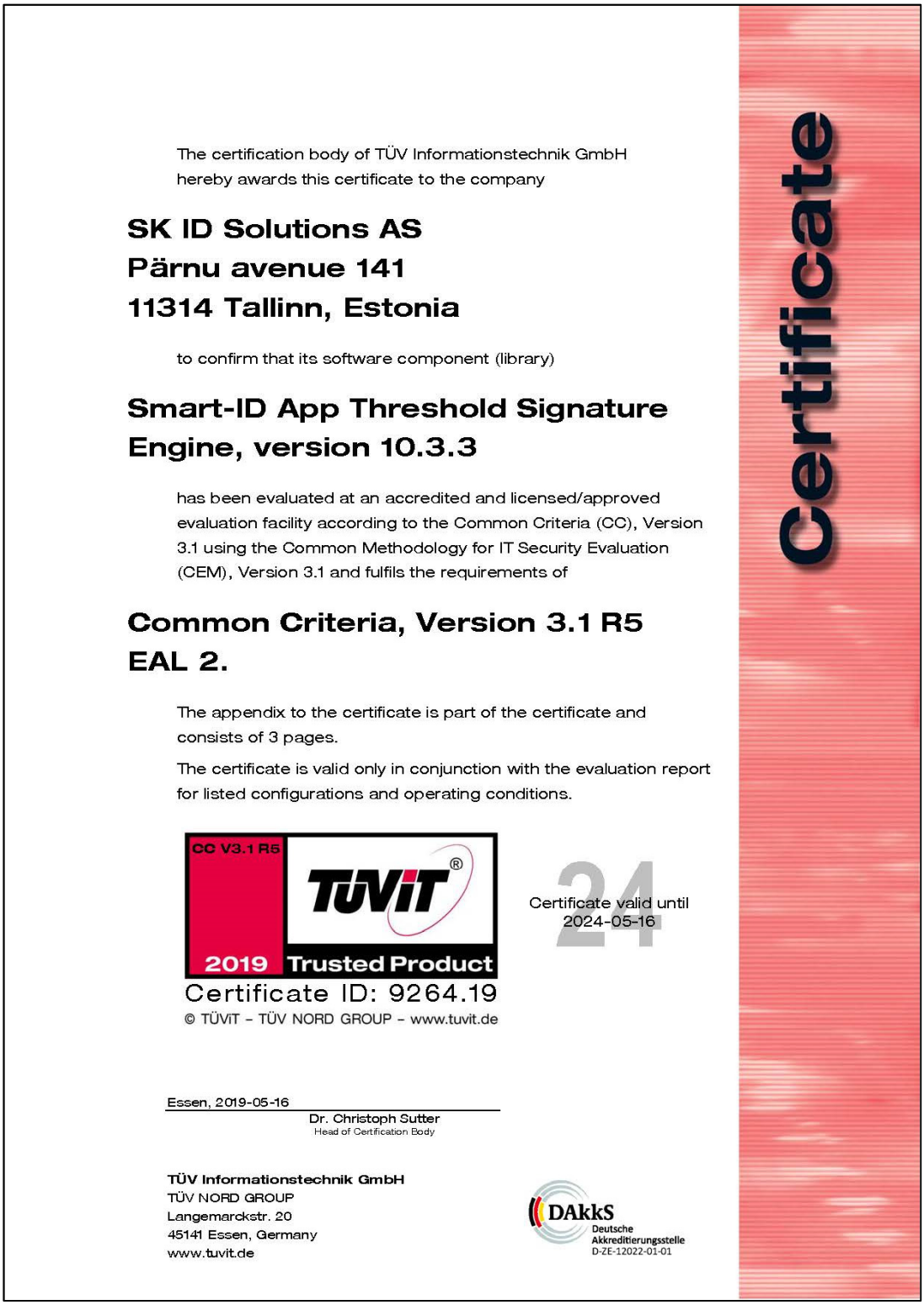
Part A

Certificate and Background of the Certification

Part A presents a copy of the issued certificate and summarizes

- information about the certification body,
- the certification procedure, and
- the performance of evaluation and certification.

1 The Certificate



Certificate

The certification body of TÜV Informationstechnik GmbH hereby awards this certificate to the company

SK ID Solutions AS
Pärnu avenue 141
11314 Tallinn, Estonia

to confirm that its software component (library)

Smart-ID App Threshold Signature Engine, version 10.3.3

has been evaluated at an accredited and licensed/approved evaluation facility according to the Common Criteria (CC), Version 3.1 using the Common Methodology for IT Security Evaluation (CEM), Version 3.1 and fulfils the requirements of

Common Criteria, Version 3.1 R5 EAL 2.

The appendix to the certificate is part of the certificate and consists of 3 pages.

The certificate is valid only in conjunction with the evaluation report for listed configurations and operating conditions.

CC V3.1 R5 TÜViT 2019 Trusted Product

Certificate ID: 9264.19

Certificate valid until 2024-05-16

Essen, 2019-05-16

Dr. Christoph Sutter
Head of Certification Body

TÜV Informationstechnik GmbH
TÜV NORD GROUP
Langemarckstr. 20
45141 Essen, Germany
www.tuvit.de

DAkkS Deutsche Akkreditierungsstelle D-ZE-12022-01-01

2 Certification Body – CERTÜViT

The Certification Body of *TÜV Informationstechnik GmbH*[1] – TÜV NORD GROUP – was established in 1998 and offers a variety of services in the context of security evaluation and validation.

TÜViT is accredited for certification of IT security products according to ITSEC and Common Criteria by *Deutsche Akkreditierungsstelle GmbH under* registration no. D-ZE-12022-01-01 and performs its projects under a quality management system certified against ISO 9001.

3 Specifications of the Certification Procedure

The certification body conducts the certification procedure according to the criteria laid down in the following:

- DIN EN ISO/IEC 17065
- TÜViT Certification Scheme
- TÜViT Certification Conditions
- Common Criteria for Information Technology Security Evaluation (CC) part 1-3, version 3.1 revision 5, April 2017.
- Common Methodology for Information Technology Security Evaluation (CEM), version 3.1 revision 5, April 2017.
- Application Notes and Interpretations of the Scheme (AIS), published by BSI.

4 Performance of Evaluation and Certification

The certification body monitors each individual evaluation to ensure uniform procedures, interpretations of the criteria, and ratings. The software component (library) Smart-ID App Threshold Signature Engine, Version 10.3.3 has undergone the certification procedure at TÜViT certification body.

The evaluation of the software component (library) Smart-ID App Threshold Signature Engine, Version 10.3.3 was conducted by the evaluation body for IT-security of TÜViT and concluded on May 16, 2019. The TÜViT evaluation body is recognised by BSI.

Sponsor as well as the developer is SK ID Solutions AS. Distributor of the product is SK ID Solutions AS.

[1] in the following termed shortly TÜViT

The certification was concluded with

- the comparability check and
- the preparation of this certification report.

This work was concluded on May 16, 2019. The confirmation of the evaluation assurance level (EAL) only applies on the condition that:

- all stipulations regarding generation, configuration and operation, as given in part B of this report, are observed,
- the product is operated – where indicated – in the environment described.

This certification report applies only to the version of the product indicated here. The validity of the certificate can be extended to cover new versions and releases of the product, provided the applicant applies for re-certification of the modified product, in accordance with the procedural requirements, and provided the evaluation does not reveal any security deficiencies.

This certificate is not an endorsement of the IT product by the TÜV Informationstechnik GmbH or any other organisation that recognises or gives effect to this certificate, and no warranty of the IT product by TÜV Informationstechnik GmbH or any other organisation that recognises or gives effect to this certificate, is either expressed or implied.

In order to avoid an indefinite usage of the certificate when evolved attack methods require a re-assessment of the products resistance to state of the art attack methods, the maximum validity of the certificate has been limited. The certificate issued on May 16, 2019 is valid until May 16, 2024. The validity date can be extended by re-assessment and re-certification.

With regard to the meaning of the evaluation assurance levels (EAL), please refer to part C of this report.

Within the last two years, the certifier did not render any consulting or other services for the company ordering the certification and there was no relationship between them that might have an influence on his assessment.

The certifier did not participate at any time in test procedures for the product, which forms the basis of the certification.

5 Publication

The following Certification Results consist of pages B-1 to B-18. The certification report and the certificate for software component (library) Smart-ID App Threshold Signature Engine, Version 10.3.3 will be included in the TÜViT certification list (http://www.tuvit.de).

Further copies of this certification report may be ordered from the sponsor of the product. The certification report may also be obtained in electronic form at the internet address of TÜViT as stated above.

Part B

Certification Result

The following results represent a summary of

- the security target of the sponsor for the target of evaluation,
- the relevant evaluation results from the evaluation facility, and
- complementary notes and stipulations of the certification body.

Contents of the Certification Result

1 Executive Summary

The target of evaluation (TOE) is the software component (library) ***Smart-ID App Threshold Signature Engine, Version 10.3.3***.

The TOE is a software component (library), which implements the client-side functions of the Threshold Signature Scheme Protocol (TSSP) (see section 2.2 of [ST]). The TOE is intended to be embedded inside an Android/iOS mobile application, which provides a GUI for the Signer (the end-user). One of such mobile application implementations is the Smart-ID App. The TOE works in co-operation with the Smart-ID backend system and the Smart-ID SecureZone component, which implements the server-side functions of the TSSP algorithm.

The TOE is intended to be used as a component of a QSCD system to support the following high-level functions:

- Creation of Qualified Electronic Signatures;
- Enrolment and destruction of the Signer's key pair;
- Security management and access control functions.

The security target [ST] is the basis of this certification. It is not based on a certified protection profile.

The TOE security assurance requirements are based entirely on the assurance components and classes defined in part 3 of Common Criteria (see part C of this report or [CC] Part 3 for details). The TOE meets the assurance requirements of assurance level EAL 2 (Evaluation Assurance Level 2).

The TOE's security functional requirements were taken from CC part 2 (i. e. the set is CC part 2 conformant) [CC]. They are implemented by the following nine security functions:

Security Function	Description
SF.SecureChannel	SF.SecureChannel ensures an encrypted communication between the TOE and the Smart-ID backend components in order to provide confidentiality and detection of modifications.
SF.CryptoAlgorithms	SF.CryptoAlgorithms enforces the use of cryptographic algorithms that ensure that signatures can not be forged.
SF.KeyGen	SF.KeyGen generates the client's key share with RSA key generation algorithm and implements the distribution of the key part.
SF.EncryptedStorage	SF.EncryptedStorage stores cryptographic key material outside of the TOE within the app's sandbox storage area encrypted with AES encryption algorithm.
SF.Signing	SF.Signing ensures the generation of the signature share.

Security Function	Description
SF.KeyZer	SF.KeyZer enforces the TOE to destroy cryptographic keys after they are no longer used.
SF.CloneDetection	SF.CloneDetection implements the Clone Detection protocol, which helps the Smart-ID SecureZone to detect, when there are two copies of the private key in use.
SF.PINQuality	SF.PINQuality verifies the user-supplied PIN against the blacklist and against the length limitation.
SF.SecurityTest	SF.SecurityTest performs tests to verify the consistency of the configuration data, the statistical quality of the environment provided PRNG and the positive authentication of the Smart-ID backend services with the HTTPS pinning.

A more detailed description of the TOE security functions can be found in section 7.2 of the public ST, which is attached as part E of this certification report.

Assets for the TOE comprise the integrity and/or confidentiality security functions of the TOE and the data used like the data to be signed representation, the electronic signature with the different shares, the signature verification data and the cryptographic keys during operation.

The six threats deal with the creation of one or more forged signatures, the change of data to be signed under the name of the signer, the decrease of trust in the signatures created with the service Smart-ID Trust Service Provider (TSP), and the security of the TOE. The threats are organised within the ST in the following subsections in order to present the closely related threats next to each other:

- Threats related to the key enrolment,
- Threats related to impersonation of the Signer within the signing process.

There are six organisational security policies for the TOE.

A more detailed description of the threats, organisational security policies and assumptions can be found in sections 4.4, 4.5 and 4.6 of the public ST, which is attached as part E of this certification report. The certification covers the configurations of the TOE as outlined in chapter 8.

2 Identification of the TOE

The Target of Evaluation (TOE) is the software component (library) Smart-ID App Threshold Signature Engine, Version 10.3.3.

The TOE delivery consists of the following parts:

1. TOE Documentation
2. Smart-ID App Threshold Signature Engine

The TOE including the TOE documentation is composed in a software zip-archive, which is delivered via a secure file transfer delivery system. The integrity of the delivered TOE has to be checked comparing the SHA-384 hash values of the TOE.

No.	Type	Item / file name / SHA-384 hash value	Note	Form of Delivery
1.	SW	Android SmartId TSE binary package (file name: smartid-tse-10.3.3_RELEASE-release.aar) 5f7a7574a737a0ed48616a60ed683aa5b4a59bd9af223e091f149c270d4f5f06b2a26e94297404443fb80949bfb83ff3	Only for Android TOE software part	Secure file transfer system
2.	SW	Android CryptoLib binary package (file name: smartid-jlib-10.3.3_RELEASE-release.aar) 3e2ca98784e38f66ba399eacebd2f42636e52e13d355e6105d3f8a9714b2967c232be6dd6215d82d978628cf170e8379	Only for Android TOE software part	Secure file transfer system
3.	SW	iOS TSE binary package (file name: libSmartIdTSE_10.3.3.a) a32be245fa8150d21458228501a85f69600665064c0a3fc09911ac0b04575b9c1e71070dec061053db81b38bb25461b8	Only for iOS TOE software part	Secure file transfer system
4.	txt	iOS TSE header package (file name: SmartIdTSE_include_10.3.3.zip) 625f10272c3df66285b58cd6523fefb59022b6f558e16d202b28b5489aec5f1d87e00c7bf78bec8ce1f8c029d6dfdef7	Only for iOS TOE software part	Secure file transfer system
5.	txt	Release notes	Part of acceptance procedure	Secure file transfer system
6.	txt	checksums	Part of acceptance procedure	Secure file transfer system
7.	DOC	Smart-ID App TSE library's integration guide for Android (file name: tse_integration_guide_Android.pdf) 7562f4adeb040f1e7ddba2525a8d615536e69f5e030078e12c0275f430a6a5c7fd24247a2ef19a4e91aa0fc1f281ed86	Only for Android Part of guidance documentation	Secure file transfer system

No.	Type	Item / file name / SHA-384 hash value	Note	Form of Delivery
8.	DOC	Smart-ID App TSE library's integration guide for iOS (file name: tse_integration_guide_iOS.pdf) d5dbed8993d06f1ae66fd010b3167a818838a28cedadeda780b85b1aff92979206e151c5904b47d0be39ed53fcd23b84	Only for iOS Part of guidance documentation	Secure file transfer system
9.	DOC	Signer User Guidance information for SecureZone and TSE library operators (file name: signers_guide.pdf) a610a5a36770e796c6f74969ecb40986fc2724b5ba55179f6f55f35ed1f54d09da7f86e77cc608abddff6f9100d08c18	Part of guidance documentation	Secure file transfer system
10.	DOC	Javadoc documentation TSE 837a19c411f177e6527caa4ca38346a1fe9a9ba0d5174a9130c1664c212d04cf0da305943d4d7fb31ec886717a4cfa55,	Only for Android part of guidance documentation	Secure file transfer system
	DOC	Javadoc documentation JLIB 324f744b6718a78ba9f747646a65a6cc82e9dc9be41ec396a9d7d43f469e7a92edc04a79f0e30e89287a3c24a21978fd	Only for Android part of guidance documentation	Secure file transfer system
11.	DOC	Headerdoc documentation 9429d8c426e3c759e8bd523cf0804bac228e4724d4b6511f93cce27787c4f463348b4be4bb7411e114dc7ce55ff40933	Only for iOS part of guidance documentation	Secure file transfer system

3 Security Policy

The security policy enforced is defined by the selected set of Security Functional Requirements and implemented by the TOE. It covers the following issues:

- Cryptographic Support,
- User Data Protection,
- Identification and Authentication,
- Protection of the TSF,

Specific details concerning the different security policies can be found in section 7 of the public ST, which is attached as part E of this certification report.

4 Assumptions and Clarification of Scope

The assumptions defined in the Security Target and some aspects of threats and Organisational Security Policies are not covered by the TOE itself. These aspects lead to

specific security objectives to be fulfilled by the TOE environment (see the following chapter in the Security Target:

- 5.2 Security Objectives for the Environment fulfilled by mobile platform

5 Architectural Information

The TOE consists of two subsystems with in total 15 modules for the Android Implementation and 14 modules for the iOS implementation:

Subsystem	Module	Description
SmartIdTSE (Android)	SmartIdTSE API module	*The SmartIdTSE API module provides an interface to the whole SmartIdTSE module. The SmartIdTSE API uses the singleton software pattern and requires initiation every time the singleton is created in device memory. The API interfaces are thread safe and will throw an exception or return an error object if they are used in the wrong order or at the wrong time based on the current state of the SmartIdTSE API module. The SmartIdTSE API module has state.*
	Key Generation Manager module	*The Key Generation Manager module provides a way to generate key material for new key pairs and to test the PRNG available on the device during key generation.*
	Key Manager module	*The Key Manager module allows to store, update and remove app's share of the private keys and related information.*
	Key Storage module	*The Key Storage module implements methods that allow to store, update and delete keys and related objects.*
	Key State Manager module	*The Key State Manager module allows to initiate and process key creation, data signing and key clone detection actions and ensure the process follows any and all requirements of the state machine.*
	Key State Runnable module	*The Key State Runnable module is part of the state machine the Key State Manager module uses to facilitate functionality of the methods confirmTransaction, submitClientSecondPart and the use of the clone detection protocol.*
	Network API module	*The Network API module allows to execute requests to Smart-ID SecureZone. It uses the Network Stack module to make use of the securely configured network stack*

Subsystem	Module	Description
	Network Stack module	*The Network Stack module provides an interface for both internal and external components to use the same securely configured HTTPS stack to run their own API implementations. The module accepts an interface and returns an implementation of the API described in the interface.*
SmartIdCryptoLib (Android)	CryptoLib API module	*The CryptoLib API module is a façade software pattern that provides access to all of the interfaces of SmartIdCryptoLib modules via a single interface. That means that all calls to the CryptoLib API are delegated to other modules for the actual execution.*
	Crypto module	*The Crypto module provides interface to encrypt, decrypt data and validate signatures.*
	Encode module	*The Encode module provides utility interfaces for encoding, decoding and converting between various data formats. It also contains methods for padding payloads before encrypting them with the Crypto module.*
	RandomGeneration module	*The RandomGeneration module provides access to device's PRNG. It also allows to run tests on the RPNG to check the quality of the output.*
	KeyGeneration module	*The KeyGeneration module provides a way to generate a new key pair. It also allows to generate Diffie-Hellman key agreement and ConcatKDF based derived keys to secure the communication with Smart-ID SecureZone.*
	Signing module	*The Signing module provides a way to sign a transaction digest with a app's part of the private key. It also allows to generate a Verification code based on a transaction digest.*
	Storage module	*The Storage module provides a way for the SmartItTSE to store and treieve objects to and from the local storage. All the storage related interfaces are key-value based and do not know anything about the actual objects being stored and retrieved.*
SmartIdTSE (iOS)	SmartIdTSE API module	*The SmartIdTSE API module provides an interface to the whole SmartIdTSE module. The SmartIdTSE API uses the singleton software pattern and requires initiation every time the singleton is created in device memory. The API interfaces are thread safe and will throw an exception or return an*

Subsystem	Module	Description
		error object if they are used in the wrong order or at the wrong time based on the current state of the SmartIdTSE API module.
	Key Manager module	*The Key Manager Module allows to store, update and remove app's share of the private keys and related information.*
	Key Storage module	*The Key Storage module implements methods that allow to store, update and delete keys and related objects.*
	Key State Manager module	*The Key State Manager module allows to initiate and process key creation, data singing and key clone detection actions and ensure the process follows any and all requirements of the state machine.*
	Key State Operation module	*The Key State Operation module allows to wrap the logic for all the state machine based actions and to use the Key State Operation module to execute them on as simultaneous operations.*
	Network API module	*The Network API module allows to execute requests to Smart-ID SecureZone. It uses the Network Stack module to make use of the securely configured network stack.*
	Network Stack module	*The Network Stack module provides an interface for both internal and external components to use the same securely configured HTTPS stack to run their own API implementations. The module accepts an interface and returns an implementation of the API described in the interface.*
SmartIdCryptoLib (iOS)	Cryption module	*The Cryption module provides interface to encrypt and decrypt data.*
	Encoding module	*The Encode module provides utility interfaces for encoding and decoding and converting between various data formats.*
	RandomGeneration module	*The RandomGeneration module provides access to device's PRNG. It also allows to run tests on the RPNG to check the quality of the output.*
	KeyAgreement module	*The Key agreement module allows creation of Diffie-Hellman keypair to be used to encrypt data between SmartIdTSE and Smart-ID SecureZone.*
	KeyGeneration module	*The KeyGeneration module provides a way to generate a new key pair.*
	Signing module	*The signing module provides a way to decrypt the app's share of the private key with the supplied PIN and using that to sign*

Subsystem	Module	Description
		a digest. It also allows to generate a verification code based on a digest.
	Storage module	*The Storage module provides a way for the SmartIdTSE to store and retrieve objects to and from the local storage. Alle the storage related interfaces are key-value based and do not know anything about the actual objects being stored and retrieved.*

6 Documentation

The following documentation is provided with the product by the developer to the consumer (see chapter 2).

7 IT Product Testing

The developer's testing to systematically test the TOE security functionality / TSFI, was executed with the following approach:

- The Tests covered two configurations of the TOE (iOS and Android) as identified in the [ST].
- Tests cover the TSFI and their behavioral aspects defined in [ADV_TDS_A] and [ADV_TDS_I], by choosing the TSFI methods to be covered with unit tests as follows:
    - The TSFI functions not covered directly by unit tests are executed by higher-level tests of Smart-ID ecosystem (e.g. at the Smart-ID App level).
    - Test have been created for TSFI with functionality that has several edge cases that are hard to test on a real device but are easy to write tests for.
    - For some TSFI there is no test directly for, because they delegate their functionality to sub functions that are covered with other tests.
    - Some TSFI deal with external artefacts (connection to SecureZone) that with the current abstraction level have no easy way to mock for tests. These TSFI were not tested directly with automated test. Where possible, tests have been added for more concrete sub functions of these TSFI.

The evaluation body testing started on December 3rd, 2018 and was successfully concluded on December 5th 2018. The evaluator's objective was to test the functionality of the TOE systematically against the security functionality description in [ST] and [ADV]. In order to do this, the evaluation body performed the following tasks:

- Repeat the developer's tests,
- Devise and execute own functional tests.

8 Evaluated Configuration

The TOE Smart-ID App Threshold Signature Engine, Version 10.3.3 is delivered in two configurations, one for the Android implementation and one for the iOS implementation.
The Security Target [ST] has identified two configurations of the TOE under evaluation.

9 Results of the Evaluation

9.1 CC specific results

The Evaluation Technical Report [ETR] was provided by TÜViT's evaluation body according to the requirements of the Scheme, the Common Criteria [CC], the Methodology [CEM] and the Application Notes and Interpretations of the Scheme [AIS].
As a result of the evaluation the verdict PASS is confirmed for the following assurance components:

- All components of the EAL2 package including the class ASE as defined in the CC (see also part C of this report).

The verdicts for CC, part 3 assurance classes and components (according to EAL2 and the class ASE for the Security Target Evaluation) are summarised in the following table:

Assurance classes and components			**Verdict**
Development		**ADV**	**PASS**
	Security architecture description	ADV_ARC.1	PASS
	Security-enforcing functional specification	ADV_FSP.2	PASS
	Basic design	ADV_TDS.1	PASS
Guidance documents		**AGD**	**PASS**
	Operational user guidance	AGD_OPE.1	PASS
	Preparative procedures	AGD_PRE.1	PASS
Life-cycle support		**ALC**	**PASS**
	Use of a CM system	ALC_CMC.2	PASS
	Parts of the TOE CM coverage	ALC_CMS.2	PASS
	Delivery procedures	ALC_DEL.1	PASS
Security Target evaluation		**ASE**	**PASS**
	Conformance claims	ASE_CCL.1	PASS
	Extended components definition	ASE_ECD.1	PASS
	ST introduction	ASE_INT.1	PASS
	Security objectives	ASE_OBJ.2	PASS

Assurance classes and components			Verdict
	Derived security requirements	ASE_REQ.2	PASS
	Security problem definition	ASE_SPD.1	PASS
	TOE summary specification	ASE_TSS.1	PASS
Tests		**ATE**	**PASS**
	Evidence of coverage	ATE_COV.1	PASS
	Functional testing	ATE_FUN.1	PASS
	Independent testing - sample	ATE_IND.2	PASS
Vulnerability Assessment		**AVA**	**PASS**
	Vulnerability analysis	AVA_VAN.2	PASS

9.2 Results of the cryptographic assessment

The strength of the cryptographic algorithms was not rated in the course of this certification procedure (see [BSIG], section 9, para. 4, clause 2). But Cryptographic Functionalities with a security level of lower than 100 bits can no longer be regarded as secure without considering the application context. Therefore, for these functionalities it shall be checked whether the related crypto operations are appropriate for the intended system. Some further hints and guidelines can be derived from [TR-02102].

Any Cryptographic Functionality that is marked with 'No' in column 'Security Level above 100 Bits' of the following table achieves a security level of lower than 100 Bits (in general context).

No.	Purpose	Cryptographic Mechanism	Implementation Standard	Key Size in Bits	Security Level above 100 Bits	Evaluator's comments
1.	**Generation of Signature verification** data to perform digital signature verification.	RSA PKCS1-v1_5	[RFC8017], TSSP	2048, 3072,	Yes	For 2047, 2048 Bits: Minimum: End 2022. For > 3000 Bits: Presently without maximum validity.
2.	**Generation of the share** of the Signer's private key.	RSA_PKCS1-v1_5	[RFC8017], TSSP	2048, 3072,	Yes	For 2048 Bits: Minimum: End 2022. For > 3000 Bits: Presently without maximum validity.
3.	Generation of symmetric encryption/decryption and integrity	Diffie-Hellman station-to-station protocol and concatKDF	[RFC2631], [RFC3526], [SP800-56A Rev.2]	2048 up to 16384	Yes	For 2048 Bits: Minimum: End 2022.

No.	Purpose	Cryptographic Mechanism	Implementation Standard	Key Size in Bits	Security Level above 100 Bits	Evaluator's comments
	protection key to create the **secure communication channel** between TOE and SecureZone.					For >3000 Bits: Presently without maximum validity.
4.	**Generation of** a part of the compound **signature** of the Signer.	RSA_PKCS1_v1_5	[RFC8017], TSSP	2047, 2048, 3071, 3072,	Yes	For 2047, 2048 Bits: Minimum: End 2022. For > 3000 Bits: Presently without maximum validity.
5.	**Secure storage:** Encryption, decryption of the key material in the sandbox storage.	AES	[FIPS 197]	128, 256	Yes	Presently without maximum validity.
6.	**Secure channel:** Perform signature validation and encryption when securing the communication between TOE and TSE library in the possession of the Signer.	RSASSA-PKCS1-V1_5 and RSAES-OAEP	[RFC8017]	2048 up to 16384	Yes	For 2048 Bits: Minimum: End 2022. For >3000 Bits: Presently without maximum validity.
7.	**Secure channel:** Message encryption and decryption between the TOE and SecureZone	AES	[FIPS 197]	128, 256	Yes	Presently without maximum validity.
8.	**Secure channel:** Provide and verify the authenticity and integrity of the messages between the specific instance of the TSE library used by the Signer and the SecureZone.	HMAC	[FIPS 198-1]	128	Yes	Presently without maximum validity.
9.	Digest computation for key generation operations.	SHA-256	[FIPS 180-4]	256	Yes	Presently without maximum validity.

10 Evaluation Stipulations, Comments, and Recommendations

The evaluation technical report contains no stipulations or recommendations.

11 Certification Stipulations and Notes

There are no stipulations or notes resulting from the certification report.

12 Security Target

The security target [ST] for *Smart-ID App Threshold Signature Engine, Version 10.3.3* is included in part E of this certification report.

13 Definitions

13.1 Acronyms

AGD	Guidance Documents
CC	Common Criteria for Information Technology Security Evaluation (referenced to as [CC])
CEM	Common Methodology for Information Technology Security Evaluation (referenced to as [CEM])
EAL	Evaluation Assurance Level
EEPROM	Electrical Erasable and Programmable Read Only Memory
ES	Embedded Software
EU	European Union
FSP	Functional Specification
FIPS	Federal Information Processing Standard
HLD	High-level Design
HSM	Hardware Security Module
IC	Integrated Circuit
JWE	JSON Web Encryption
JSON	Java Script Object Notation
IF	Interface
IGS	Installation, Generation and Start-up
OS	Operating System
OSP	Organisational Security Policy
PP	Protection Profile
PRNG	Pseudo-random number generator
RPC	Remote Procedure Call
RSA	Signature Algorithm of Rivest, Shamir, Adleman
SAR	Security Assurance Requirement
SF	Security Function
SFP	Security Function Policy
SFR	Security Functional Requirement
SIF	Sub-interface
SOF	Strength of Function
SS	Sub-system
SSL	Secure Sockets Layer
ST	Security Target
TOE	Target of Evaluation
TSC	TSF Scope of Control
TSE	Threshold Signature Engine
TSF	TOE Security Functions

TSFI	TOE Security Function Interfaces
TSP	TOE Security Policy
TSSP	Threshold Signature Scheme Protocol
VLA	Vulnerability Analysis

13.2 Glossary

Augmentation	The addition of one or more assurance component(s) from Part3 to an EAL or assurance package.
Extension	The addition to an ST or PP of functional requirements not contained in Part 2 and/or assurance requirements not contained in Part 3 of the CC.
Formal	Expressed in a restricted syntax language with defined semantics based on well-established mathematical concepts.
Informal	Expressed in natural language.
Object	An entity within the TSC that contains or receives information and upon which subjects perform operations.
Protection Profile	An implementation-independent set of security requirements for a category of TOEs that meet specific consumer needs.
Security Function	A part or parts of the TOE that have to be relied upon for enforcing a closely related subset of the rules from the TSP.
Security Target	A set of security requirements and specifications to be used as the basis for evaluation of an identified TOE.
Semiformal	Expressed in a restricted syntax language with defined semantics.
Strength of Function	A qualification of a TOE security function expressing the minimum efforts assumed necessary to defeat its expected security behaviour by directly attacking its underlying security mechanisms.
Subject	An entity within the TSC that causes operations to be performed.
Target of Evaluation	An IT product or system and its associated administrator and user guidance documentation that is the subject of an evaluation.
TOE Security Functions	A set consisting of all hardware, software, and firmware of the TOE that must be relied upon for the correct enforcement of the TSP.
TOE Security Policy	A set of rules that regulate how assets are managed, protected and distributed within a TOE.
TSF Scope of Control	The set of interactions that can occur with or within a TOE and are subject to the rules of the TSP.

14 Bibliography

[ADV_TDS_A] Target of Evaluation (TOE) Basic Design for Android, Version 2.2.0 as of 2018-10-15

[ADV_TDS_I] Target of Evaluation (TOE) Basic Design for iOS, Version 2.4.1, as of 2019-10-23

[AGD_PRE_A] Smart-ID App TSE library's Integration guide for Android, Version 2.2.4 as of 2018-10-26

[AGD_PRE_I] Smart-ID App TSE library's Integration guide for iOS, Version 2.4.5 as of 2018-10-26

[AGD_OPE] Signer User Guidance information for SecureZone and TSE library operators, Version 2.3_v27 as of 2018-10-03

[AIS] Application Notes and Interpretations of the Scheme (AIS) as relevant for the TOE, Bundesamt für Sicherheit in der Informationstechnik

[AIS1] Anwendungshinweise und Interpretationen zum Schema (AIS), AIS 1, Durchführung der Ortsbesichtigung in der Entwicklungsumgebung des Herstellers, Version 13, 2008-08-14, Bundesamt für Sicherheit in der Informationstechnik.

[AIS11] Anwendungshinweise und Interpretationen zum Schema (AIS), AIS 11, Programmiersprachen und Compiler, Version 2.0, 1998-02-02, Bundesamt für Sicherheit in der Informationstechnik.

[AIS14] Anwendungshinweise und Interpretationen zum Schema, AIS 14: Anforderungen an Aufbau und Inhalt der ETR-Teile (Evaluation Technical Report) für Evaluationen nach CC (Common Criteria), Version 7, 2010-08-03, Bundesamt für Sicherheit in der Informationstechnik.

[AIS19] Anwendungshinweise und Interpretationen zum Schema (AIS), AIS 19, Anforderungen an Aufbau und Inhalt der Zusammenfassung des ETR (Evaluation Technical Report) für Evaluationen nach CC (Common Criteria), Version 9, 2014-11-03, Bundesamt für Sicherheit in der Informationstechnik.

[AIS32] Anwendungshinweise und Interpretationen zum Schema (AIS), AIS 32, CC-Interpretationen im deutschen Zertifizierungsschema, Version 7, 2011-06-08, Bundesamt für Sicherheit in der Informationstechnik.

[AIS40] Application Notes and Interpretation of the Scheme (AIS), AIS 40, Use of Interpretation for Security Evaluation and Certification of Digital Tachographs, Version 1, 2005-06-28, Bundesamt für Sicherheit in der Informationstechnik.

[AIS41] Application Notes and Interpretation of the Scheme (AIS) – AIS 41, Guidelines for PPs and STs, Version 2, 2011-01-31, Bundesamt für Sicherheit in der Informationstechnik.

[AIS42] Application Notes and Interpretation of the Scheme (AIS), AIS 42, Guidelines for the Developer Documentation, Version 1, 2008-05-21, Bundesamt für Sicherheit in der Informationstechnik.

[CC] Common Criteria for Information Technology Security Evaluation, Version 3.1,
Part 1: Introduction and general model, Revision 5, April 2017
Part 2: Security functional requirements, Revision 5, April 2017
Part 3: Security assurance requirements, Revision 5, April 2017

[CEM] Common Methodology for Information Technology Security Evaluation, Evaluation Methodology, Version 3.1, Revision 5, April 2017

[eIDAS] 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.

[ETR] Evaluation Technical Report, TÜV Informationstechnik GmbH, version *2*, *2019-05-16*, project-number: 8114700517

[ST] Smart-ID App Threshold Signature Engine Security Target , Version 2.4.2 as of 2018-10-09, SK ID Solutions AS

[TR-02102] BSI - Technische Richtlinie TR-02102 Kryptographische Verfahren: Empfehlungen und Schlüssellängen (consisting of [TR-02102-1]/[TR-02102-2]/[TR-02102-3])

[TR-02102-1] BSI - Technische Richtlinie TR-02102-1, Kryptographische Verfahren: Empfehlungen und Schlüssellängen, Version 2014-01, 2014-02-10, Bundesamt für Sicherheit in der Informationstechnik.

[TR-02102-2] BSI - Technische Richtlinie TR-02102-2, Kryptographische Verfahren: Empfehlungen und Schlüssellängen, Teil 2 – Verwendung von Transport Layer Security (TLS), Version 2014-01, 2014-02-12, Bundesamt für Sicherheit in der Informationstechnik.

[TR-02102-3] BSI - Technische Richtlinie TR-02102-3, Kryptographische Verfahren: Empfehlungen und Schlüssellängen, Teil 3 – Verwendung von Internet Protocol Security (IPsec) und Internet Key Exchange (IKEv2), Version 2014-01, 2014-02-12, Bundesamt für Sicherheit in der Informationstechnik.

Part C
Excerpts from the Criteria

The excerpts from the criteria are dealing with

- conformance results
- assurance categorization
- evaluation assurance levels
- strength of security function
- vulnerability analysis

CC Part 1:

Conformance Claim

The conformance claim indicates the source of the collection of requirements that is met by a PP or ST that passes its evaluation. This conformance claim contains a CC conformance claim that:

- describes the version of the CC to which the PP or ST claims conformance.
- describes the conformance to CC Part 2 (security functional requirements) as either:

  CC Part 2 conformant - A PP or ST is CC Part 2 conformant if all SFRs in that PP or ST are based only upon functional components in CC Part 2, or

  CC Part 2 extended - A PP or ST is CC Part 2 extended if at least one SFR in that PP or ST is not based upon functional components in CC Part 2.
- describes the conformance to CC Part 3 (security assurance requirements) as either:

  CC Part 3 conformant - A PP or ST is CC Part 3 conformant if all SARs in that PP or ST are based only upon assurance components in CC Part 3, or

  CC Part 3 extended - A PP or ST is CC Part 3 extended if at least one SAR in that PP or ST is not based upon assurance components in CC Part 3.

Additionally, the conformance result may include a statement made with respect to sets of defined requirements, in which case it consists of one of the following:

- Package name Conformant - A PP or ST is conformant to a pre-defined package (e. g. EAL) if:
  - the SFRs of that PP or ST are identical to the SFRs in the package, or
  - the SARs of that PP or ST are identical to the SARs in the package.
- Package name Augmented - A PP or ST is an augmentation of a pre-defined package if:
  - the SFRs of that PP or ST contain all SFRs in the package, but have at least one additional SFR or one SFR that is hierarchically higher than an SFR in the package.
  - the SARs of that PP or ST contain all SARs in the package, but have at least one additional SAR or one SAR that is hierarchically higher than an SAR in the package

Note that when a TOE is successfully evaluated to a given ST, any conformance claims of the ST also hold for the TOE. A TOE can therefore also be e. g. CC Part 2 conformant.

Finally, the conformance result may also include a statement made with respect to Protection Profiles, in which case it includes the following:

- **PP Conformant** - A PP or TOE meets specific PP(s), which are listed as part of the conformance result.
- **Conformance Statement** (Only for PPs) - This statement describes the manner in which PPs or STs must conform to this PP: strict or demonstrable. For more information on this Conformance Statement, see Annex D.

CC Part 3:

Class APE: Protection Profile evaluation

Evaluating a PP is required to demonstrate that the PP is sound and internally consistent, and, if the PP is based on one or more other PPs or on packages, that the PP is a correct instantiation of these PPs and packages. These properties are necessary for the PP to be suitable for use as the basis for writing an ST or another PP.

Assurance Class	Assurance Components
Class APE: Protection Profile evaluation	APE_INT.1 PP introduction
	APE_CCL.1 Conformance claims
	APE_SPD.1 Security problem definition
	APE_OBJ.1 Security objectives for the operational environment APE_OBJ.2 Security objectives
	APE_ECD.1 Extended components definition
	APE_REQ.1 Stated security requirements APE_REQ.2 Derived security requirements

APE: Protection Profile evaluation class decomposition

Class ACE: Protection Profile Configuration evaluation

Evaluating a PP-Configuration is required to demonstrate that the PP-Configuration is sound and consistent. These properties are necessary for the PP-Configuration to be suitable for use as the basis for writing an ST or another PP or PP-Configuration.

Assurance Class	Assurance Components
Class ACE: Protection Profile Configuation evaluation	ACE_INT.1 PP-Module introduction
	ACE_CCL.1 PP-Module conformance claims
	ACE_SPD.1 PP-Module Security problem definition
	ACE_OBJ.1 PP-Module Security objectives
	ACE_ECD.1 PP-Module extended components definition
	ACE_REQ.1 PP-Module security requirements
	ACE_MCO.1 PP-Module consistency
	ACE_CCO.1 PP-Configuration consistency

APE: Protection Profile Configuration evaluation class decomposition

Class ASE: Security Target evaluation

Evaluating an ST is required to demonstrate that the ST is sound and internally consistent, and, if the ST is based on one or more PPs or packages, that the ST is a correct instantiation of these PPs and packages. These properties are necessary for the ST to be suitable for use as the basis for a TOE evaluation.

Assurance Class	Assurance Components	
Class ASE: Security Target evaluation	ASE_INT.1	ST introduction
	ASE_CCL.1	Conformance claims
	ASE_SPD.1	Security problem definition
	ASE_OBJ.1	Security objectives for the operational environment ASE_OBJ.2 Security objectives
	ASE_ECD.1	Extended components definition
	ASE_REQ.1 ASE_REQ.2	Stated security requirements Derived security requirements
	ASE_TSS.1 ASE_TSS.2	TOE summary specification TOE summary specification with architectural design summary

ASE: Security Target evaluation class decomposition

Security assurance components

“The following Sections describe the constructs used in representing the assurance classes, families, and components.“ “Each assurance class contains at least one assurance family.” “Each assurance family contains one or more assurance components.”

The following table shows the assurance class decomposition:

Assurance Class	Assurance Components	
ADV: Development	ADV_ARC.1	Security architecture description
	ADV_FSP.1	Basic functional specification
	ADV_FSP.2	Security-enforcing functional specification
	ADV_FSP.3	Functional specification with complete summary
	ADV_FSP.4	Complete functional specification
	ADV_FSP.5	Complete semi-formal functional specification with additional error information

Assurance Class	Assurance Components
	ADV_FSP.6 Complete semi-formal functional specification with additional formal specification
	ADV_IMP.1 Implementation representation of the TSF ADV_IMP.2 Implementation of the TSF
	ADV_INT.1 Well-structured subset of TSF internals ADV_INT.2 Well-structured internalsignV_INT.3 Minimally complex internals
	ADV_SPM.1 Formal TOE security policy model
	ADV_TDS.1 Basic design ADV_TDS.2 Architectural design ADV_TDS.3 Basic modular design ADV_TDS.4 Semiformal modular design ADV_TDS.5 Complete semiformal modular design ADV_TDS.6 Complete semiformal modular design with formal high-level design presentation
AGD: Guidance documents	AGD_OPE.1 Operational user guidance
	AGD_PRE.1 Preparative procedures

Assurance Class	Assurance Components
ALC: Life cycle support	ALC_CMC.1 Labelling of the TOE ALC_CMC.2 Use of a CM system ALC_CMC.3 Authorisation controls ALC_CMC.4 Production support, acceptance procedures and automation ALC_CMC.5 Advanced support
	ALC_CMS.1 TOE CM coverage ALC_CMS.2 Parts of the TOE CM coverage ALC_CMS.3 Implementation representation CM coverage ALC_CMS.4 Problem tracking CM coverage ALC_CMS.5 Development tools CM coverage
	ALC_DEL.1 Delivery procedures
	ALC_DVS.1 Identification of security measures ALC_DVS.2 Sufficiency of security measures
	ALC_FLR.1 Basic flaw remediation ALC_FLR.2 Flaw reporting procedures ALC_FLR.3 Systematic flaw remediation
	ALC_LCD.1 Developer defined life-cycle mode ALC_LCD.2 Measurable life-cycle mode
	ALC_TAT.1 Well-defined development tools ALC_TAT.2 Compliance with implementation standards ALC_TAT.3 Compliance with implementation standards - all parts

Assurance Class	Assurance Components
ATE Tests	ATE_COV.1 Evidence of coverage ATE_COV.2 Analysis of coverage ATE_COV.3 Rigorous analysis of coverage
	ATE_DPT.1 Testing: basic design ATE_DPT.2 Testing: security enforcing modules ATE_DPT.3 Testing: modular design ATE_DPT.4 Testing: implementation representation
	ATE_FUN.1 Functional testing ATE_FUN.2 Ordered functional testing
	ATE_IND.1 Independent testing – conformance ATE_IND.2 Independent testing – sample ATE_IND.3 Independent testing – complete
AVA: Vulnerability assessment	AVA_VAN.1 Vulnerability survey AVA_VAN.2 Vulnerability analysis AVA_VAN.3 Focused vulnerability analysis AVA_VAN.4 Methodical vulnerability analysis AVA_VAN.5 Advanced methodical vulnerability analysis

Assurance class decomposition

Evaluation assurance levels

The Evaluation Assurance Levels (EALs) provide an increasing scale that balances the level of assurance obtained with the cost and feasibility of acquiring that degree of assurance. The CC approach identifies the separate concepts of assurance in a TOE at the end of the evaluation, and of maintenance of that assurance during the operational use of the TOE.

It is important to note that not all families and components from Part 3 are included in the EALs. This is not to say that these do not provide meaningful and desirable assurances. Instead, it is expected that these families and components will be considered for augmentation of an EAL in those PPs and STs for which they provide utility.

Evaluation assurance level (EAL) overview

The above table represents a summary of the EALs. The columns represent a hierarchically ordered set of EALs, while the rows represent assurance families. Each number in the resulting matrix identifies a specific assurance component where applicable.
As outlined in the next section, seven hierarchically ordered evaluation assurance levels are defined in the CC for the rating of a TOE's assurance. They are hierarchically ordered

inasmuch as each EAL represents more assurance than all lower EALs. The increase in assurance from EAL to EAL is accomplished by *substitution* of a hierarchically higher assurance component from the same assurance family (i. e. increasing rigour, scope, and/or depth) and from the *addition* of assurance components from other assurance families (i. e. adding new requirements).
These EALs consist of an appropriate combination of assurance components as described in chapter 2 of CC Part 3. More precisely, each EAL includes no more than one component of each assurance family and all assurance dependencies of every component are addressed.
While the EALs are defined in the CC, it is possible to represent other combinations of assurance. Specifically, the notion of “augmentation“ allows the addition of assurance components (from assurance families not already included in the EAL) or the substitution of assurance components (with another hierarchically higher assurance component in the same assurance family) to an EAL. Of the assurance constructs defined in the CC, only EALs may be augmented. The notion of an “EAL minus a constituent assurance component“ is not recognised by the CC as a valid claim. Augmentation carries with it the obligation on the part of the claimant to justify the utility and added value of the added assurance component to the EAL. An EAL may also be extended with explicitly stated assurance requirements.

Evaluation assurance level 1 (EAL1) - functionally tested

EAL1 is applicable where some confidence in correct operation is required, but the threats to security are not viewed as serious. It will be of value where independent assurance is required to support the contention that due care has been exercised with respect to the protection of personal or similar information.

EAL 1 requires only a limited security target. It is sufficient to simply state the SFRs that the TOE must meet, rather than deriving them from threats, OSPs and assumptions through security objectives.

EAL 1 provides an evaluation of the TOE as made available to the customer, including independent testing against a specification, and an examination of the guidance documentation provided. It is intended that an EAL 1 evaluation could be successfully conducted without assistance from the developer of the TOE, and for minimal outlay.
An evaluation at this level should provide evidence that the TOE functions in a manner consistent with its documentation.

Evaluation assurance level 2 (EAL2) - structurally tested

EAL2 requires the co-operation of the developer in terms of the delivery of design information and test results, but should not demand more effort on the part of the developer than is consistent with good commercial practice. As such it should not require a substantially increased investment of cost or time.
EAL2 is therefore applicable in those circumstances where developers or users require a low to moderate level of independently assured security in the absence of ready availability of the

complete development record. Such a situation may arise when securing legacy systems, or where access to the developer may be limited.

Evaluation assurance level 3 (EAL3) - methodically tested and checked

EAL3 permits a conscientious developer to gain maximum assurance from positive security engineering at the design stage without substantial alteration of existing sound development practices.
EAL3 is applicable in those circumstances where developers or users require a moderate level of independently assured security, and require a thorough investigation of the TOE and its development without substantial re-engineering.

Evaluation assurance level 4 (EAL4) - methodically designed, tested, and reviewed

EAL4 permits a developer to gain maximum assurance from positive security engineering based on good commercial development practices which, though rigorous, do not require substantial specialist knowledge, skills, and other resources. EAL4 is the highest level at which it is likely to be economically feasible to retrofit to an existing product line.
EAL4 is therefore applicable in those circumstances where developers or users require a moderate to high level of independently assured security in conventional commodity TOEs and are prepared to incur additional security-specific engineering costs.

Evaluation assurance level 5 (EAL5) - semiformally designed and tested

EAL5 permits a developer to gain maximum assurance from security engineering based upon rigorous commercial development practices supported by moderate application of specialist security engineering techniques. Such a TOE will probably be designed and developed with the intent of achieving EAL5 assurance. It is likely that the additional costs attributable to the EAL5 requirements, relative to rigorous development without the application of specialised techniques, will not be large.
EAL5 is therefore applicable in those circumstances where developers or users require a high level of independently assured security in a planned development and require a rigorous development approach without incurring unreasonable costs attributable to specialist security engineering techniques.

Evaluation assurance level 6 (EAL6) - semiformally verified design and tested

EAL6 permits developers to gain high assurance from application of security engineering techniques to a rigorous development environment in order to produce a premium TOE for protecting high value assets against significant risks.
EAL6 is therefore applicable to the development of security TOEs for application in high risk situations where the value of the protected assets justifies the additional costs.

Evaluation assurance level 7 (EAL7) - formally verified design and tested

EAL7 is applicable to the development of security TOEs for application in extremely high risk situations and/or where the high value of the assets justifies the higher costs. Practical application of EAL7 is currently limited to TOEs with tightly focused security functionality that is amenable to extensive formal analysis.

Assurance Class	Assurance Family	Assurance Components by Evaluation Assurance Level						
		EAL1	EAL2	EAL3	EAL4	EAL5	EAL6	EAL7
Development	ADV_ARC		**1**	1	1	1	1	1
	ADV_FSP	**1**	**2**	**3**	**4**	**5**	5	**6**
	ADV_IMP				**1**	1	**2**	2
	ADV_INT					**2**	**3**	3
	ADV_SPM						**1**	1
	ADV_TDS		**1**	**2**	**3**	**4**	**5**	**6**
Guidance documents	AGD_OPE	**1**	1	1	1	1	1	3
	AGD_PRE	**1**	1	1	1	1	1	1
Live cycle support	ALC_CMC	**1**	**2**	**3**	**4**	4	**5**	5
	ALC_CMS	**1**	**2**	**3**	**4**	**5**	5	5
	ALC_DEL		**1**	1	1	1	1	1
	ALC_DVS			**1**	1	1	**2**	*2*
	ALC_FLR							
	ALC_LCD			**1**	1	1	1	**2**
	ALC_TAT				**1**	**2**	**3**	3
Security Target Evaluation	ASE_CCL	**1**	1	1	1	1	1	1
	ASE_ECD	**1**	1	1	1	1	1	1
	ASE_INT	**1**	1	1	1	1	1	1
	ASE_OBJ	**1**	**2**	2	2	2	2	2
	ASE_REQ	**1**	**2**	2	2	2	2	2
	ASE_SPD		**1**	1	1	1	1	1
	ASE_TSS	**1**	1	1	1	1	1	1
Tests	ATE_COV		**1**	**2**	2	2	**3**	3
	ATE_DPT			**1**	1	**3**	3	4
	ATE_FUN		**1**	1	1	1	**2**	2
	ATE_IND	**1**	**2**	2	2	2	2	**3**
Vulnerability Assessment	AVA_VAN	**1**	**2**	2	**3**	**4**	**5**	5

Evaluation assurance level summary

Class AVA: Vulnerability assessment

The AVA: Vulnerability assessment class addresses the possibility of exploitable vulnerabilities introduced in the development or the operation of the TOE.

Vulnerability analysis (AVA_VAN)

Vulnerability analysis is an assessment to determine whether potential vulnerabilities identified, during the evaluation of the development and anticipated operation of the TOE or by other methods (e. g. by flaw hypotheses or quantitative or statistical analysis of the security behaviour of the underlying security mechanisms), could allow attackers to violate the SFRs.

Vulnerability analysis deals with the threats that an attacker will be able to discover flaws that will allow unauthorised access to data and functionality, allow the ability to interfere with or alter the TSF, or interfere with the authorised capabilities of other users.

Part D

Evaluation Results regarding development and production environment

The IT product Smart-ID App Threshold Signature Engine, Version 10.3.3 has been evaluated at an approved evaluation facility using the Common Methodology for IT Security Evaluation (CEM) Version 3.1 extended by Scheme Interpretations, by advice of the Certification Body for components beyond EAL 5 and CC Supporting documents for conformance to the Common Criteria for IT Security Evaluation (CC), Version 3.1.

As a result of the TOE certification dated 2019-05-16 the following results regarding the development and production environment apply. (ALC_CMC.2, ALC_CMS.2, ALC_DEL.1) are fulfilled for the development and production sites of the TOE listed below:

Name of site / Company name	**Address**	**Type of site**	**Date of last audit**	**New audit / reused audit / n.r.**
Tallinn, Estonia	SK ID Solutions AS Pärnu mnt 141, 11314 Tallinn, Estonia	TOE development (implementation and testing), TOE production and delivery initiation (TOE distribution), Development of CC evaluation evidence documentation.	2018-07-09/10	new audit

For the site listed above, the requirements have been specifically applied in accordance with the Security Target [ST]. The evaluators verified, that the threats, security objectives and requirements for the TOE life cycle phases up to delivery as stated in the Security Target [ST] are fulfilled by the procedures of this site.

Part E

Security Target

Attached is the public version of the Security Target: “Smart-ID App Threshold Signature Engine Security Target, Author: SK ID Solutions AS
Date: 2018-10-09
Version: 2.4.2

Smart-ID App Threshold Signature Engine Security Target

Technical document
Version 2.4.2
October 9, 2018
58 pages

Contents

List of Figures

List of Tables

1 Introduction

1.1 Objectives and Scope of the Document

This document presents the Common Criteria (CC) Security Target (ST) document for the Threshold Signature Engine component (herein referenced as TSE) of the Smart-ID system. The ST defines the Target of Evaluation (TOE) and describes the security problem with the terms of CC.

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 SecureZone Security Target*, version 2.7.0, 2018.

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

[9] *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*, FIPS SP 800-22r1a, NIST, Apr. 2010. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf.

[10] *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.

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

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

[13] *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.

[14] *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.

[15] *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.

[16] *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.

[17] *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 Abbrevations

Notation	Description
CA	Certificate Authority – see also Certificate Service Provider (CSP).
CC	Common Criteria
CSP	Certificate Service Provider – service, which issues the certificates binding together the SVD and identity of Signer. See also Certificate Authority (CA).

Notation	Description
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.
ICT	Information and Communications Technology
JWE	JSON Web Encryption – data structure representing an encrypted and integrity-protected message.
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.
MITM	Man in the Middle– Attack where the transmitted data is modified and/or replayed by attacker eavesdropping in the middle of transmitter and receiver.
PKI	Public Key Infrastructure
QES	Qualified Electronic Signature – an electronic signature that is compliant to the reg. (EU) 910/2014 [4] for electronic transactions within the internal European market.
QSCD	Qualified Signature Creation Device – device, which produces the Qualified Electronic Signature (QES) according to the reg. (EU) 910/2014 [4].
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.
Signer	The natural person, who is the owner of the key pair (Signature Creation Data (SCD) and Signature Verification Data (SVD)) and who is creating the digital signatures with the key pair.
SSCD	Secure Signature Creation Device
ST	Security Target
SVD	Signature Verification Data – the public key corresponding to the SCD for a signature, which can be used to verify the signature.
SZ	SecureZone – Smart-ID SecureZone is the software component, which implements the server-side functions of the Threshold Signature Scheme Protocol (TSSP) and provides services for the key enrolment and signature creation.
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

Notation	Description
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 TSSP and to use the Smart-ID SecureZone services for the key enrolment and signature creation.
TSF	TOE Security Functions
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

1.5 ST Reference Identification

Title: Smart-ID App Threshold Signature Engine Security Target
Version: 2.4.2
Publication date: 09 October 2018

1.6 TOE Reference Identification

TOE identification: Smart-ID App Threshold Signature Engine
TOE version: v10.3.3

1.7 Document changelog

Version	Date	Summary of changes
1.0.0	04.06.2017	First submission to the evaluation process
1.0.1	11.07.2017	1. Fixed the problems outlined on the "Observation Report V1", observations 1 to 39. Detailed list of individual changes are listed in the response to the report. 2. Deleted the Appendix, which contained out-dated and no longer useful information.

Version	Date	Summary of changes
2.0.0	02.02.2018	1. Rewrite of the document to be more similar with the concepts of PP 419 241-2 [6]. 2. Aligned the document with the modified SecureZone ST document version 2.0.0 [7].
2.1.0	14.05.2018	Corrected the errors and problems with the observations 1, 44 to 75
2.2.0	22.06.2018	Fixed the problems outlined in the Observation Report V3. Detailed list of individual changes are listed in the response to the report.
2.3.0	08.08.2018	Fixed the problems outlined in the Observation Report V5. Detailed list of individual changes are listed in the response to the report.
2.3.1	30.08.2018	Removed the FDP_SDI.1, which was erroneously used to describe the Verification Code feature.
2.4.0	25.09.2018	Updated the used key lengths defined in SFRs FCS_COP.1/SHA-2, FCS_COP.1.1/AES, FCA_COP.1/HMAC, FCS_CKM.1.1/RSA_ClientShare to correspond with the implementation. Updated the wording of FCS_COP.1.1/RSA_Other to also reflect the encryption/decryption operation with RSA keys.
2.4.1	28.09.2018	Updated the description of physical scope of the TOE.
2.4.2	09.10.2018	Fixed the wording of supported Android/iOS operating system versions in section 2.3.4.

2 System Overview

This chapter provides an informal overview of the digital signatures, Smart-ID Threshold Signature Scheme Protocol and defines Smart-ID App Threshold Signature Engine component as the Target of Evaluation (TOE). 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 and the main security features of the TOE, the components of the TOE environment and the TOE intended usage are described.

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 App Threshold Signature Engine (TSE), which is the client-side implementation of this protocol and the Target of Evaluation (TOE) of this ST document.

2.2 Threshold Signature Scheme Protocol (TSSP)

To better explain the TOE security features and the functions within the Smart-ID System, firstly the TSSP is described. Note that this section is the same as the section 2.3 in the SZ ST

document [7], but it is written from the viewpoint of the TOE (TSE) of the current ST document with slightly changed wordings. The section is repeated here for the reader's convenience.

2.2.1 Introduction

The TSSP is the protocol to be followed by the TSE and the SecureZone (SZ), in order to generate the key pair of the Signer (the assets D.SCD and D.SVD), which is usable only when Signer, TSE and the SecureZone 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 TSE (the asset D.clientPart) and the SecureZone (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 SecureZone. 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, TSE 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 [8] in order to get the implementation details of the TOE.

2.2.2 Key pair generation process

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

2.2.2.1 Actors and components

- Signer – This is the natural person, who is using the Smart-ID App TSE and the SecureZone 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 (TOE) – 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 are evaluated according to the current ST document.
- Smart-ID SecureZone – This is the software component, which is the TOE of the SecureZone ST document [7]. The SecureZone implements the server-side functions of the TSSP. The SZ 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 SZ to store user data. Sensitive security attributes are stored with Hardware Security Module (HSM) proprietary encryption or with SZ implemented encryption.
- Smart-ID SecureZone HSM – This is the trusted hardware component, which is providing the certified cryptographic functions to the SZ, such as key share generation and creation of the signature share.

2.2.2.2 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.
7. TSE generates the app's share of the key pair (D.clientShare/D.clientModulus). The key pair consists of the private key (D.clientShare) 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 (D.clientShare) of the generated key pair into two parts (D.clientPart/D.serverPart), 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 D.clientShare 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 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.

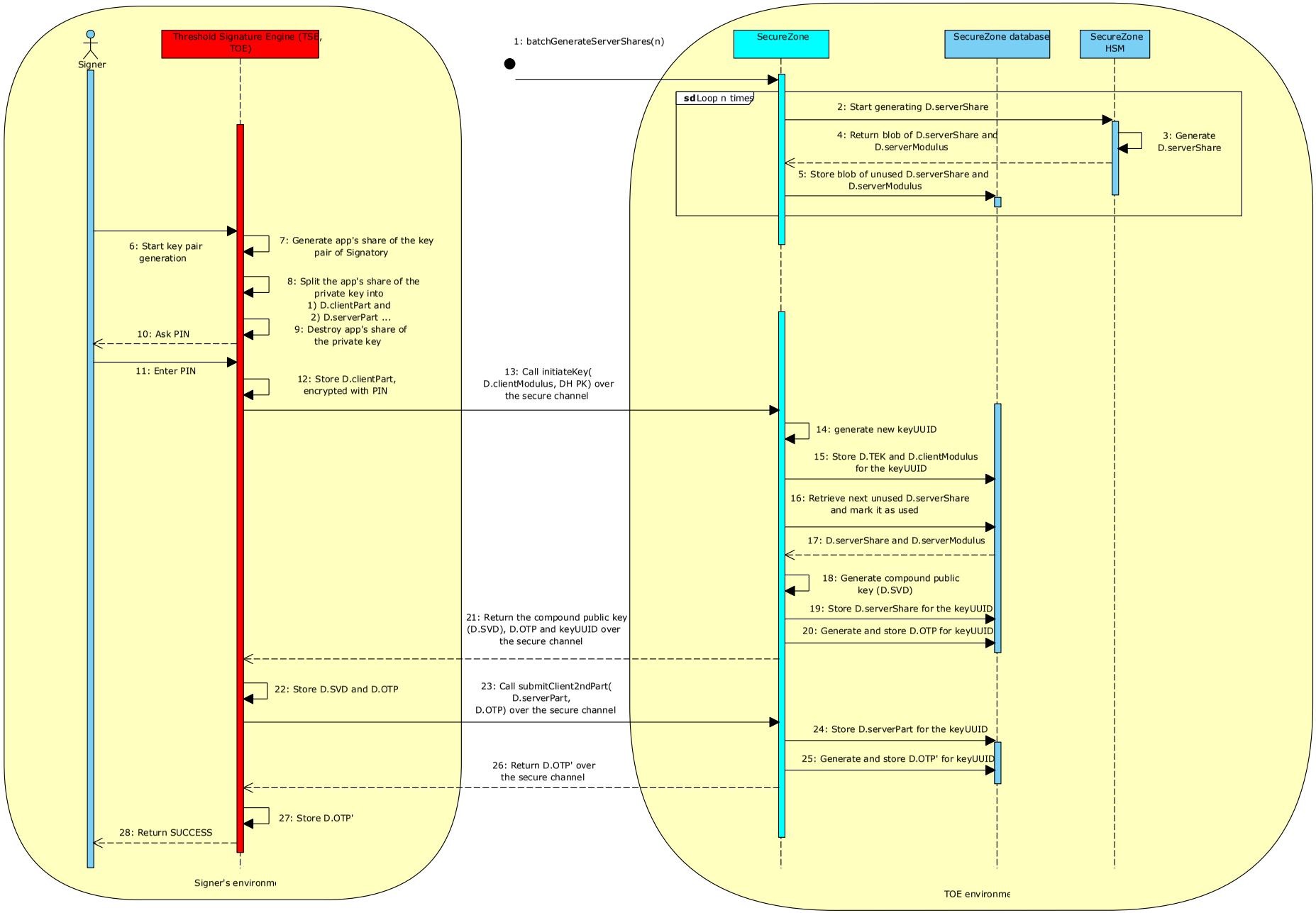


Figure 1. Overview of the enrollment procedure in the TSSP.

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 in the SZ, 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.

2.2.3 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 (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 SZ itself, in a similar way as the function for creation of digital documents are not included in the SSCDs.

2.2.3.1 Actors and components

- Signer – This is the natural person, who is using the SCA, Smart-ID App TSE and the SZ 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 Data To Be Signed Representation (DTBS/R) and requesting the digital signature of the DTBS/R from the Signer.
- Smart-ID App TSE (TOE) – 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.

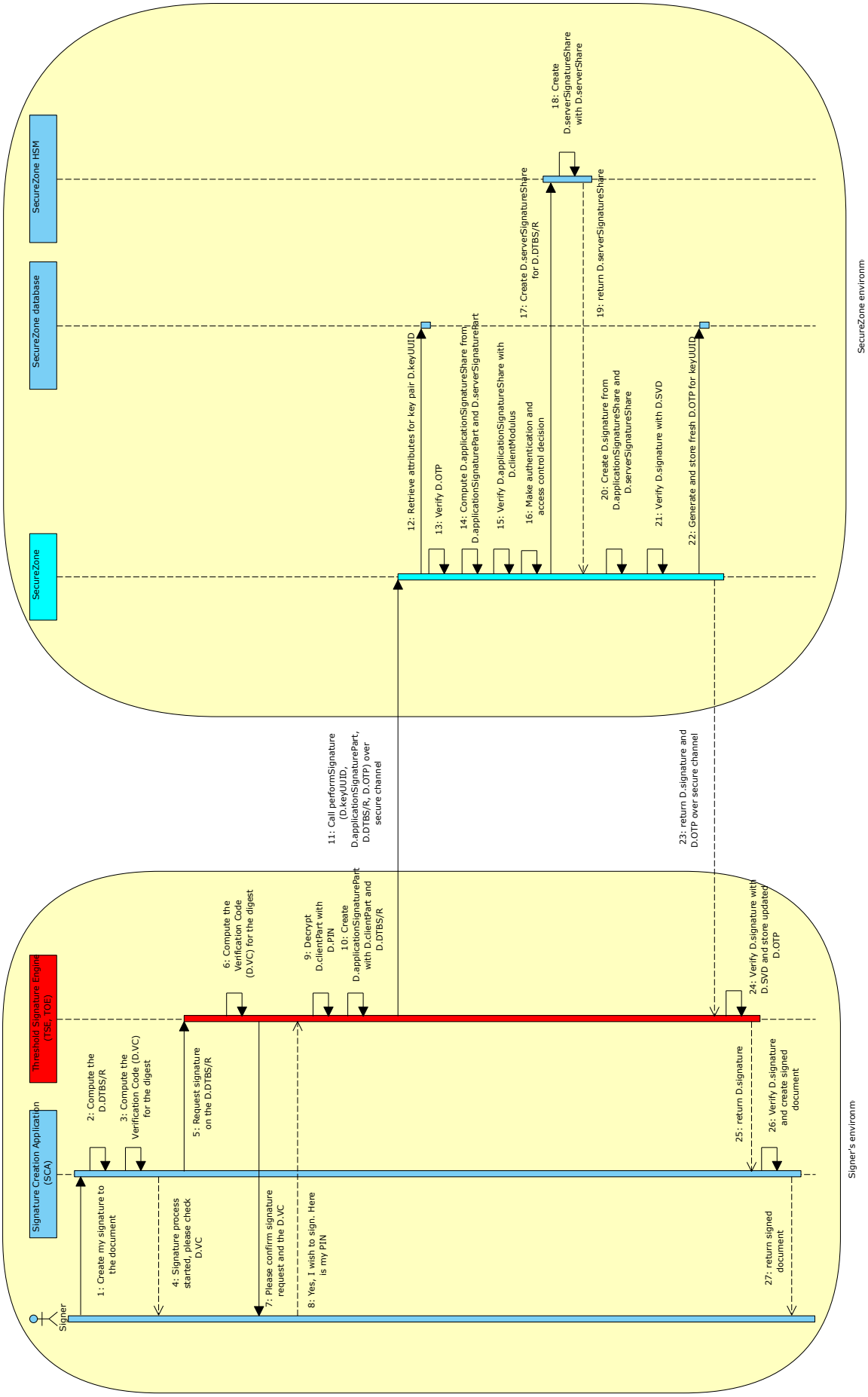


Figure 2. Overview of the signing procedure in the TSSP.

- Smart-ID SecureZone – This is the software component, which is the TOE of the SecureZone ST document [7]. The SecureZone implements the server-side functions of the TSSP. The SZ 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 SZ to store user data and TSF data. Sensitive security attributes are stored with HSM proprietary encryption or with SZ implemented encryption.
- Smart-ID SecureZone HSM – This is the hardware component, which is providing the certified cryptographic functions to the SZ, such as key share generation and signature share generation.

2.2.3.2 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.
6. TSE receives the request and computes the verification code (D.VC).
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 SZ.
10. TSE uses the decrypted D.clientPart to create the signature share D.applicationSignature Part with the D.DTBS/R.
11. TSE initiates the SZ performSignature() operation over the secure channel, along with the following data: D.Signing_Key_Id, D.applicationSignaturePart, D.DTBS/R, D.OTP.
12. SZ retrieves attributes for the keypair D.Signing_Key_Id from the database.
13. SZ verifies that clone-detection tokens (D.OTP) for that particular D.Signing_Key_Id are valid. This gives us the possession-based authentication factor.
14. SZ 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. SZ verifies if the signature share D.applicationSignatureShare is valid, with the D.clientModulus.

16. SZ 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. SZ 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. SZ receives the D.serverSignatureShare and verifies it with D.serverModulus and D.DTBS/R.
20. SZ creates the compound signature D.signature from the signature shares D.application SignatureShare and D.serverSignatureShare.
21. SZ verifies, if the compound signature D.signature is valid with the D.DTBS/R and D.SVD.
22. SZ generates fresh clone-detection tokens (D.OTP) and stores them in database.
23. SZ returns the compound signature D.signature and updated D.OTP to the TSE over the secure channel.
24. TSE decrypts the response 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 (this is not shown in figure 2) 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.

2.3 Overview of the TOE

This section describes the TOE and explains its intended usage.

2.3.1 TOE definition

The Smart-ID App TSE is a software library, which implements the client-side functions of the TSSP to enable the creation of electronic signatures on behalf of the Signer.

The TOE is intended to be embedded inside an Android/iOS mobile application which provides a GUI for the Signer (the end-user). One of such mobile application implementations is the Smart-ID App.

The TOE works in co-operation with the Smart-ID backend system and the Smart-ID SecureZone component, which implements the server-side functions of the TSSP algorithm.

2.3.2 TOE type

The TOE is a software component, which implements the client-side functions of the TSSP.

It is deployed in a Signer's mobile device and connects to the Smart-ID SecureZone component over the secure channel. It prepares the Signature Activation Data (SAD), which is required to complete the signature computation with the SecureZone and HSM.

Together, the Smart-ID App TSE TOE, the Smart-ID SecureZone component, and the HSM, are a Qualified Signature Creation Device (QSCD).

2.3.3 TOE usage and major security functions

2.3.3.1 TOE usage

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

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

2.3.3.2 Major security functions

The TOE performs critical security and cryptography functions of the client-side implementation of TSSP algorithm.
The main security functions of the TOE are as follows:

1. Generation of D.clientShare, verification that the TOE environment meets the security requirements before the key generation.
2. Splitting the private exponent of D.clientShare into the D.clientPart and D.serverPart
3. Key registration with the Smart-ID SecureZone component
4. Secure handling and destruction of the key material
5. Computing the D.applicationSignaturePart
6. Secure handling of the user's PIN-code and ensuring that the chosen PIN code meets the security requirements
7. Participating in the Clone Detection algorithm to enable detection of cloned mobile devices
8. Establishing secure communication channel with the Smart-ID backend system, including the Smart-ID SecureZone component

The TOE provides assurances that the D.clientShare is under sole control of the Signer and that the Signer intends to sign the D.DTBS/R.

2.3.4 Required non-TOE hardware/software/firmware

The TOE requires the following hardware and software to run:

1. Mobile device which is running Android/iOS operating system. The supported Android operating system versions are 4.1 or higher, the supported iOS versions are 8.0 and higher.
2. Mobile application, which has implemented the UI and interfaces with the TOE API. The mobile application has to embed the TOE. One of such applications is Smart-ID App.
3. Smart-ID backend system along with Smart-ID SecureZone component, which provide the server-side functions of the Smart-ID system.

2.3.5 Physical scope of the TOE

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

1. TSE library's binary file(s) of the specified platform, delivered within library file(s)
2. Guidance documentation for TSE library of the specified platform, consisting of:

2.1 Smart-ID App TSE library's integration guide for iOS/Android (in .pdf format)

2.2 TSE library's API documentation - Headerdoc for iOS/Javadoc for Android platform (in compressed format)

2.3 Signer User Guidance information for SecureZone and TSE library operators (in .pdf format). The document contains information addressed to the TSE library's integrator with the guidance on how to write operational instructions for the end-user (the signer).

2.3.6 Logical scope of the TOE

This section describes the logical scope of the TOE.

2.3.6.1 Key pair generation and enrolment

TOE performs the client-side functions of the TSSP algorithm and implements the key pair generation and enrolment procedures as shown in the section 2.2.2.

2.3.6.2 Signature computation

TOE implements the client-side functions of the signature computation procedure as shown in the section 2.2.3.

2.3.6.3 Protecting communication with external components

1. Secure channel with external components - TOE uses JSON Web Encryption (JWE) messages for communicating with the Smart-ID SecureZone. JWE messages are encrypted with the D.TEK and they are integrity protected.

2.3.7 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, certificate issuance and other features, which are commonly required by the full-scale PKI system.

Other features, which may be provided by the Smart-ID App as well, are not included in the logical scope of the TOE, for example:

1. Functions required for Signer registration, authentication and Signer identity proofing,
2. Functions for managing the mobile application's UI.

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] extended and Common Criteria Part 3 [3] conformant.

3.2 Package conformance

This ST conforms to assurance package EAL2 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.DTBS_Integrity

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.applicationSignature Share can be validated with D.clientModulus.	confidentiality, integrity
D.Authorisation_ Data	It is the data used by the SecureZone to authorise the signature computation. The D.Authorisation_Data is part of the D.SAD. The SecureZone verifies the D.SAD before the signature computation. D.Authorisation_Data consists of D.Signing_Key_Id, D.DTBS/R and D.applicationSignaturePart. The D.Authorisation_Data is created by the TSE during the key pair operation.	confidentiality, integrity
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

Name	Description	Security
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.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	Key Transfer Key (KTK). Asymmetric encryption/decryption key pair, which is used to wrap the key material during the transmission from TSE to SecureZone. SZ uses the HSM to generate and protect the private key. The embedded copy of the public key of D.KTK is included within the configuration file of the TSE.	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 SZ and never transmitted on the network. 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
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 SecureZone 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

Name	Description	Security
D.Reference_ Signer_ Authentication_ Data	This is the set of data used by SecureZone to authenticate the signer. It contains all the data and keys used by the SecureZone 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 SecureZone. D.SAD is prepared and transmitted by TSE. D.SAD consists of: 1. D.Reference_Signer_Authentication_Data, 2. D.Authorisation_Data	confidentiality, integrity
D.SCD	Signature Creation data (SCD). In the conventional digital signature systems, this corresponds to the private key of the Signer's 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.	
D.server SignaturePart	Share of the signature of D.DTBS/R, which is computed by the SecureZone 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 SecureZone and stored in the encrypted form.	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

Name	Description	Security
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.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 TSE and SecureZone use the asset D.Signing_Key_Id to identify the D.clientPart in the TSE, D.serverPart in the SZ database and D.serverShare in the Cryptographic Module. D.Signing_Key_Id is referenced as Key Universally Unique IDentifier (keyUUID) in some places since this is the name of this attribute in the developer documents and source code.	integrity
D.SVD	SVD. 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	Transport Encryption Key (TEK). Symmetric cryptographic key shared between the SecureZone 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.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 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.Integrator – Integrator is the developer of the Smart-ID App (U.Smart-ID_Service and U.UI), which uses the TOE interfaces and which embeds the TOE.

4.2.2 External IT Systems

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

1. U.Smart-ID_Service – The IT component uses the TOE interfaces and provides the more abstract level interfaces to the App UI.
2. U.UI – The IT component uses the interfaces provided by U.Smart-ID_Service and in turn, indirectly uses the TOE interfaces as well.

4.2.3 Subjects

The TOE is not authenticating the U.User by itself. Instead, the TOE is gathering all the necessary pieces to perform the multi-factor authentication with the Smart-ID SecureZone. Therefore, the TOE is not completing the binding of the U.User to any subject and the subjects are not modelled in this ST. Only reference subjects from the ST of the SZ are provided.

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 by the SecureZone 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 instance of the TSE within the mobile device of the U.User. TSE is using technical SZ functions (such as update clone detection attributes, perform re-key) on behalf of the U.User, the S.App has limited access to the SZ objects. U.User is bound to the S.App by the SecureZone after the successful single-factor authentication, which includes the possession-based information from the mobile device.

4.2.4 Roles

The TOE is not authenticating the U.User by itself, therefore any roles are not modelled in this ST document.

4.3 Threat Agents

1. U.Attacker – Attacker's main goal is to get the signature of the Signer, which the Signer has not intended to sign. Attacker can try to obtain the physical access to the TOE and the assets stored by TOE. Attacker can eavesdrop on the communication channel between the TOE and SecureZone and submit bogus information to the TOE and SecureZone. Attacker cannot read the RAM of the TOE. Attacker has the basic 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).

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

4.4.1.1 T.MITM

Attacker may try to perform Man in the Middle (MITM) attack and eavesdrop on the communication channel between the TOE and SecureZone and try to read, modify and replay the messages between the TOE and SecureZone.

The impacted assets are D.serverPart, D.Signing_Key_Id, D.clientModulus, D.application SignaturePart, D.Authorisation_Data, D.SAD, D.SVD, D.DTBS/R, D.OTP.

4.4.1.2 T.SHARE_GUESS

Attacker may try to use the brute force or the cryptographic weakness in the key generation algorithms and guess the D.clientPart, D.serverPart, D.clientShare.

4.4.1.3 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 component of 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 component of D.SCD or other encryption/decryption keys and then impersonate the Signer to the SecureZone or create Signer's signature on the fresh D.DTBS/R without Signer's consent.

The assets D.Random, D.clientPart, D.serverPart, D.clientShare, D.TEK 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

4.4.2.1 T.PIN_GUESS

Attacker may try to steal the mobile device with the TOE (usually under the control of the Signer) and try to guess the D.PIN of the Signer and then use the TOE to create signatures under name of the Signer.

The asset D.PIN is threatened.

4.4.2.2 T.PHYS_TAMPER

Attacker may try to steal the mobile device with the TOE (usually under the control of the Signer) and try to physically dissect the device and read the contents of the mobile device persistent storage and then use the revealed information to submit the signature completion call to SecureZone and create signatures under name of the Signer.

The assets, which could be attacked this way are D.clientPart, D.OTP, D.TEK.

4.4.2.3 T.CLONE

Attacker may covertly try to make the clone of the mobile device storage and after succeeding with the T.PIN_GUESS or T.SHARE_GUESS, he may try to use his clear-text copy of the D.clientPart and D.OTP.

The assets, which could be attacked this way are D.clientPart, D.OTP, D.TEK.

4.4.3 Relations between threats and assets

Table 4. Compiled overview of relations between threats and assets

Asset	Security Requirement	Threats
D.clientPart	confidentiality, integrity	T.CLONE, T.SHARE_GUESS, T.PHYS_TAMPER, T.Random
D.serverPart	confidentiality, integrity	T.MITM, T.SHARE_GUESS, T.Random
D.OTP	confidentiality, integrity	T.MITM, T.PHYS_TAMPER, T.CLONE
D.applicationSignaturePart	confidentiality, integrity	T.MITM
D.Authorisation_Data	confidentiality, integrity	T.MITM
D.clientModulus	integrity	T.MITM
D.clientShare	confidentiality, integrity	T.SHARE_GUESS, T.Random
D.DTBS/R	integrity	T.MITM

Table 4. Compiled overview of relations between threats and assets

Asset	Security Requirement	Threats
D.KTK	confidentiality, integrity	
D.PIN	confidentiality, integrity	T.PIN_GUESS
D.Random	confidentiality, integrity	T.Random
D.SAD	confidentiality, integrity	T.MITM
D.Signing_Key_Id	integrity	T.MITM
D.TEK	confidentiality, integrity	T.Random, T.PHYS_TAMPER, T.CLONE
D.VC	integrity	
D.SVD	integrity	T.MITM

4.5 Organization Security Policies

The following security policies apply to the TSE. Because TSE generates and processes some components of D.SCD, TSE has to fulfill the requirements of the reg. (EU) 910/2014 [4] and also the requirements OE.TSE.* from the SZ ST document [7]. The OE.TSE.* requirements are subsets of more general reg. (EU) 910/2014 [4] and therefore, they are listed in the same subsections.

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)).

This also covers the OE.TSE.SCD_Confidential from [7]:

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

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)).

This also covers the OE.TSE.SCD_Unique from [7]:

> 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.

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)).

This also covers the OE.TSE.Sig_Secure from [7]:

> 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.

4.5.4 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).

This also covers the OE.TSE.DTBS_Intend from [7]:

> 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.

4.5.5 P.TSSP_End2End

The [7] also lists the OE.TSE.TSSP_End2End:

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

4.5.6 P.App_Sandbox

The [7] also lists the OE.TSE.App_Sandbox:

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

4.6 Assumptions

4.6.1 A.Vigilant_User

It is assumed that the Signer follows the security best practices for the mobile devices and doesn't reveal the confidential data, such as PIN, to the attacker. Also, the Signer provides the physical security for the mobile device and keeps the mobile device under the sole control of Signer. TOE operates in a relatively secure environment, which is provided by the Signer.

4.6.2 A.Sandbox

It is assumed that the mobile platform, on top of which the Smart-ID App and the TOE is running, provides the isolation features for the memory and private, persistent storage of the apps. Attacker, who may have its own potentially harmful apps running on the Signatory's mobile device, cannot read the RAM and the persistent storage of the TOE. Signer is running authentic copy of the Smart-ID app.

4.6.3 A.CSPRNG

It is assumed that the mobile platform provides the secure random number generator, which can be used by the TOE to generated the random nonces and cryptographic keys. It is required that the random number generator satisfies the minimal set of statistical tests from the suite SP 800-22r1a [9].

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.PREVENT_BF

TOE doesn't store any reference points, i.e. the corresponding public modulus to the D.clientPart nor the D.applicationSignaturePart. In case attacker gets hold of the encrypted D.clientPart, he is not able to launch exchaustive off-line brute-force attack to try all possible combinations of D.PINs.

5.1.2 OT.SECURE_CHANNEL

TOE uses the D.KTK and D.TEK to encrypt communication to the Smart-ID SecureZone. Additionally, TOE uses HTTPS certificate pinning to authenticate the secure channel to the Smart-ID Core.

5.1.3 OT.SECURE_CRYPTO

TOE uses the well-known cryptographic algorithms to generate the keypairs and to perform other cryptographic functions.

5.1.4 OT.VERIFICATION_CODE

TOE computes the verification code for the D.DTBS/R and the Signatory is able to verify, what is the context of the transaction and which digest he is about to sign.

5.1.5 OT.ENC_STORAGE

TOE uses the encryption key derived from the Signer's D.PIN to encrypt/decrypt the locally stored D.clientPart.

5.1.6 OT.CLONE_DETECTION

TOE follows the Clone Detection protocol with the Smart-ID SecureZone and helps SZ to detect, when there are two copies of the private keys in use.

5.2 Security Objectives for the Environment

5.2.1 OE.CSPRNG

The mobile platform must provide the cryptographically secure random number generator for the TOE. TOE automatically tests, if the provided PRNG satisfies the statistical tests of the CSPRNG and in case the tests fail, the TOE refuses to initialise.

> Application Note 1
>
> The environment objective OE.CSPRNG has been defined to accurately reflect the implementation, where the TSE is using the operating system provided random number generation libraries and seeds and it is not implementing the random number generation on its own. On IOS platforms, the TSE is using the IOS standard library function SecRandomCopyBytes, which in turn uses the FIPS 140-2 [10] certified random number generation kernel module. On Android platforms, the TSE is using the Java standard library function SecureRandom, which in turn uses the Linux kernel random number generation module `/dev/random`. TSE is performing the minimal set of statistical tests from the suite SP 800-22r1a [9] on the random number generator output, to ensure the suitable quality.

5.2.2 OE.Sandbox

The mobile platform must provide the isolation features for the memory and persistent storage of the apps. Attacker, who may have its own potentially harmful apps, running on the Signatory's mobile device, cannot read the RAM and the persistent storage of the TOE.

5.2.3 OE.Vigilant_User

The user of the TOE must follow the best security practices for his/her mobile device.

5.3 Security Objectives Rationale

5.3.1 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 37,
2. mapping to environment security objectives is shown in the table 6 on page 37.

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

	OT.PREVENT_BF	OT.SECURE_CHANNEL	OT.SECURE_CRYPTO	OT.VERIFICATION_CODE	OT.ENC_STORAGE	OT.CLONE_DETECTION
T.MITM		X				
T.PIN_GUESS	X					
T.PHYS_TAMPER	X				X	X
T.SHARE_GUESS			X			X
T.CLONE						X
P.Sig_unForgeable			X			
P.SCD_Unique			X			
P.SCD_Confidential	X	X	X		X	X
P.DTBS_Integrity				X		
P.TSSP_End2End		X				
P.App_Sandbox					X	

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

	OE.CSPRNG	OE.Sandbox	OE.Vigilant_User
T.SHARE_GUESS	X		
T.Random	X		
P.Sig_unForgeable	X		
P.SCD_Unique	X		
P.SCD_Confidential	X		
P.App_Sandbox		X	
A.CSPRNG	X		
A.Sandbox		X	
A.Vigilant_User			X

5.3.1.1 Rationale for mitigating threats

T.MITM is mitigated by the using of the secure channel between the TOE and the Smart-ID SecureZone. The connection is relayed by the Smart-ID Core, which proxies the requests and responses between the TOE and SZ. The secure channel is implemented by the OT.SECURE_CHANNEL on several layers:

1. The TOE is using HTTPS communication channel to communicate with the Smart-ID Core component.
2. The TOE is authenticating the Smart-ID Core with the X.509 certificate pinning.
3. The TOE is using D.KTK in order to encrypt specific sensitive data, which is meant to be delivered directly to the Smart-ID SecureZone component.
4. The TOE is using D.TEK to encrypt and decrypt the messages between the TSE and the Smart-ID SecureZone component.

T.PIN_GUESS is mitigated by OT.PREVENT_BF. This prevents the off-line brute-force attack by not storing the PIN at all and by not providing the reference point for the attacker to validate, which PIN is correct and which is not.

T.PHYS_TAMPER is mitigated by OT.ENC_STORAGE and OT.PREVENT_BF. The first encrypts the D.clientPart so that attacker is not able to get the clear-text copy of the D.clientPart. The second prevents the off-line brute-force attack by not storing the PIN at all and by not providing the reference point for the attacker to validate, which PIN is correct and which is not. Also, in case the attacker makes the copy of user's mobile device, eavesdrops the Signer's D.PIN and then successfully deciphers the storage, the OT.CLONE_DETECTION allows the Smart-ID SecureZone to close the user's account, once the two copies of the private keys are detected.

T.CLONE is mitigated by the OT.CLONE_DETECTION, which allows the Smart-ID SecureZone to close the user's account, once the two copies of the private keys are detected.

T.SHARE_GUESS is mitigate by the OT.SECURE_CRYPTO and OE.CSPRNG. The first provides the well-known cryptographic algorithms for generating the key pairs and the second provides the secure source for the randomness. Also, the OT.CLONE_DETECTION allows the Smart-ID SecureZone to close the user's account, once the two copies of the private keys are detected.

T.Random is mitigated by OE.CSPRNG. This requires that environment provides cryptographically secure random number generator.

5.3.1.2 Rationale for fulfilling organisational policy requirements

P.Sig_unForgeable is satisfied by the OT.SECURE_CRYPTO and OE.CSPRNG. The first provides the well-known cryptographic algorithms for generating the key pairs and the second provides the secure source for the randomness.

P.SCD_Unique is satisfied by the OT.SECURE_CRYPTO and OE.CSPRNG, which provide the well-known cryptographic algorithms for generating the keypairs and secure source of randomness.

P.SCD_Confidential is satisfied by following objectives:

1. OT.SECURE_CRYPTO provides the well-known cryptographic algorithms for generating the keypairs and assuring that it is not possible to derive the D.SCD from the public D.SVD.
2. OT.PREVENT_BF prevents the off-line brute-force attack by not storing the PIN at all and by not providing the reference point for the attacker to validate, which PIN is correct and which is not.

3. OT.SECURE_CHANNEL uses the D.KTK and D.TEK to encrypt communication to the Smart-ID SecureZone.
4. OT.ENC_STORAGE assures the protection of the D.clientPart while at rest. The shares of the private key are encrypted with the key derived from the user entered D.PIN for each transaction.
5. OT.CLONE_DETECTION assures that multiple copies of leaked private keys are detected and the further damage is prevented.
6. OE.CSPRNG provides the secure source of randomness.

P.DTBS_INTEGRITY is satisfied by the OT.VERIFICATION_CODE, which is computed from the D.DTBS/R and is displayed to the Signatory by the Smart-ID App. Signatory can verify the transaction context.

P.TSSP_End2End is satisfied by the OT.SECURE_CHANNEL, which uses the D.KTK and D.TEK to encrypt communication to the Smart-ID SecureZone. Additionally, it uses HTTPS certificate pinning to authenticate the secure channel to the Smart-ID Core.

P.App_Sandbox is satisfied by the OE.SANDBOX and OT.ENC_STORAGE. OE.SANDBOX uses the mobile platform, which provide the isolation features for the memory and persistent storage of the apps. OT.ENC_STORAGE assures the protection of the D.clientPart while at rest.

5.3.1.3 Rationale for fulfilling assumptions

A.CSPRNG is satisfied by the OE.CSPRNG, which provides secure source of randomness. TOE automatically verifies the quality of random generator.

A.Sandbox is satisfied by the OE.Sandbox, which provides the isolation and protection for the persistent storage and the process memory of the Smart-ID App.

A.Vigilant_User is satisfied by the OE.Vigilant_User, which makes sure that the user of the TOE is following the best security practices.

6 Extended components definition (ASE_ECD)

6.1 Class FPT: Protection of the TSF

The additional family FPT_CLD (Clone Detection) of the Class FPT (Protection of the TOE Security Functions (TSF)) is defined here to describe the IT security functional requirement of the TOE. The TOE shall follow the clone detection protocol steps when communicating with the backend part of the Smart-ID system. Additionally, the TOE shall perform periodic NOP requests to the Smart-ID backend, so that the clones may be detected sooner. Other families within the Class FPT do not cover such kind of security features.

6.1.1 Clone Detection (FPT_CLD)

Family behaviour:
This family defines the requirements to follow the clone detection protocol, which enables the Smart-ID backend to detect, when there are multiple copies of the private key shares in use and to limit the attacker's damage.

Component levelling:

FPT_CLD.1 has two constituents:

1. FPT_CLD.1.1 – Follow Clone Detection Protocol, requires that the TOE uses the previously supplied one-time passwords and other tokens for each new subsequent Smart-ID backend API call.
2. FPT_CLD.1.2 – Periodic Clone Detection Updates, requires that the TOE implements a timer-based maintenance loop, where it performs periodic Smart-ID backend API calls, without specific user actions.

Management: FPT_CLD.1
There are no management activities foreseen.

Audit: FPT_CLD.1
There are no actions defined to be auditable.

6.1.1.1 FPT_CLD.1 – Clone Detection

Hierarchical to: No other components.
Dependencies: No dependencies.

FPT_CLD.1.1	The TOE shall use [assignment: *list of types of TSF data*] as the new one-time password for performing subsequent API call.

FPT_CLD.1.2	The TOE shall perform periodical updates of the [assignment: *list of types of TSF data*].

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 family FDP and the environment (OE.Sandbox).

Table 7. User data attributes in the TOE

Attribute name	Corresponding asset	Storage location	Notes
DTBS/R	D.DTBS/R	in memory only	The digest for the signing.

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 8 are considered 'authentication data' as per the definition of the CC Part 2, page 21, paragraph 40. Authentication data is used to create the signature and send it to Smart-ID backend when user requesting services from a TOE. The authentication data itself is protected with the SFRs from the family FPT and Encrypted Storage (Sandbox).

Table 8. Authentication data attributes in the TOE

Attribute name	Corresponding asset	Storage location	Notes
clientPart	D.clientPart	In sandbox storage, in encrypted form	Generated by the TSE and stored in the encrypted storage (Sandbox).
client_share_2nd_part	D.serverPart	Ephemeral, in memory	This is the other half of the D.clientShare. It is used to complete the signature share D.applicationSignatureShare. It is destroyed after transmission it to the SecureZone.
client_modulus	D.clientModulus	In sandbox storage	This is the public key of the D.clientShare key pair. It is used to verify the signature share D.applicationSignatureShare.
composite_modulus	D.SVD	In sandbox storage	Computed by the TOE and stored in the TOE database
OTP	D.OTP	In sandbox storage	This is the next one-timepassword, which is expected to be sent by TSE for the next key pair operation.
PIN	D.PIN	Ephemeral, in memory	PIN is known by Signer and is entered to the TSE by Signer to authorise each signing operation. The value 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.
keypairUUID	D.Signing_Key_Id	In sandbox storage	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.

Table 9. Security attributes in the TOE

Attribute name	Corresponding asset	Storage location	Notes
TEK_symmetric_key	D.TEK	In sandbox storage	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.
KTK_wrapper_key	D.KTK	In TSE configuration file	Public key part of D.KTK is included within the configuration file of the TSE which is used to wrap the key material during the transmission from TSE to SecureZone.
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 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.

7.2.1 Cryptographic support (FCS)

7.2.1.1 Cryptographic key management (FCS_CKM)

7.2.1.1.1 FCS_CKM.1/RSA_clientShare – Cryptographic key generation

First of all, TOE generates the D.clientShare.

FCS_CKM.1.1/RSA_ clientShare	The TSF shall generate **D.clientShare** ~~cryptographic keys~~ in accordance with a specified cryptographic key generation algorithm *TSSP share generation algorithm*[a] and specified cryptographic key sizes *2048, 3072 bits*[b] that meet the following: *the standard RFC8017 [11] (section 3.1) and article [5]*[c]

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

7.2.1.1.2 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 SecureZone and the 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~~ in accordance 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 [12], RFC3526 [13] and SP 800-56A Rev. 2 [14] (section 5.8.1)*[c].

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

7.2.1.1.3 FCS_CKM.4 – Cryptographic key destruction

TOE uses same key destruction method for all kind of keys:

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 [10]*[b].

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

7.2.1.2 Cryptographic operation (FCS_COP)

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

7.2.1.2.1 FCS_COP.1/RSA_SCD – Cryptographic operation

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

FCS_COP.1.1/RSA_SCD	The TSF shall perform *RSA signature creation*[a] in accordance with a specified cryptographic algorithm *TSSP signature share creation*[b] and cryptographic key sizes *2047, 2048, 3071, 3072 bits*[c] that meet the following: *standard RFC8017 [11] (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

7.2.1.2.2 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 TSE and SZ. TOE uses the algorithms in RFC8017 [11] 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 [11] (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

7.2.1.2.3 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 or 256 bits*[c] that meet the following: *standard FIPS 197 [15]*[d].

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

7.2.1.2.4 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 [16]*[d].

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

7.2.1.2.5 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 [17]*[d].

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

7.2.2 User Data Protection (FDP)

7.2.2.1 Data Authentication (FDP_DAU)

7.2.2.1.1 FDP_DAU.1 – Basic Data Authentication

FDP_DAU.1.1	The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of *D.DTBS/R*[a].

[a] assignment: list of objects or information types

FDP_DAU.1.2	The TSF shall provide *S.Signer*[a] with the ability to verify evidence of the validity of the indicated information.

[a] assignment: list of subjects

7.2.3 Identification and authentication (FIA)

7.2.3.1 Specification of secrets (FIA_SOS)

7.2.3.1.1 FIA_SOS.1 – Verification of secrets

FIA_SOS.1.1	The TSF shall provide a mechanism to verify that secrets meet *the restriction on the complexity and the length of the D.PIN*[a].

[a] assignment: a defined quality metric

7.2.4 Protection of the TSF (FPT)

7.2.4.1 Fail secure (FPT_FLS)

7.2.4.1.1 FPT_FLS.1 – Failure with preservation of secure state

FPT_FLS.1.1	The TSF shall preserve a secure state when the following types of failures occur: 1. configuration consistency verification during the start-up, 2. *failure of the PRNG.*[a]

[a] assignment: list of types of failures in the TSF

7.2.4.2 Confidentiality of exported TSF data (FPT_ITC)

7.2.4.2.1 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.

7.2.4.3 Integrity of exported TSF data (FPT_ITI)

7.2.4.3.1 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

7.2.4.4 Clone Detection (FPT_CLD)

7.2.4.4.1 FPT_CLD.1 – Clone Detection

FPT_CLD.1.1	The TSF shall use *D.OTP*[a] as the new one-time password for performing subsequent API call.

[a] assignment: list of types of TSF data

FPT_CLD.1.2	The TOE shall perform periodical updates of the *D.OTP*[a].

[a] assignment: list of types of TSF data

7.2.4.5 Testing of external entities (FPT_TEE)

7.2.4.5.1 FPT_TEE.1 – Testing of external entities

FPT_TEE.1.1	The TSF shall run a suite of tests *during the appropriate operations*[a] to check the fulfillment of 1. statistical quality of the environment provided PRNG, 2. *positive authentication of the Smart-ID Core with the HTTPS pinning.*[b]

[a] selection: during initial start-up, periodically during normal operation, at the request of an authorised user, [assignment: other conditions] [b] assignment: list of properties of the external entities

FPT_TEE.1.2	If the test fails, the TSF shall *abort the key generation or the connection to the Smart-ID Core*[a].

[a] assignment: action(s)

7.3 Security Requirements Rationale

7.3.1 Mapping between SFRs and TOE Security Objectives

The mapping of TOE Security Objectives to SFRs is shown in the table 10.

Table 10. Mapping between TOE security objectives and SFRs

	OT.PREVENT_BF	OT.SECURE_CHANNEL	OT.SECURE_CRYPTO	OT.VERIFICATION_CODE	OT.ENC_STORAGE	OT.CLONE_DETECTION
FCS_CKM.1/RSA_clientShare			X			
FCS_CKM.1/DH_TEK		X	X			
FCS_CKM.4			X			
FCS_COP.1/RSA_SCD	X	X	X		X	
FCS_COP.1/RSA_Other	X	X	X		X	
FCS_COP.1/AES	X	X	X		X	
FCS_COP.1/HMAC	X	X	X		X	
FCS_COP.1/SHA-2	X	X	X		X	
FPT_CLD.1						X
FDP_DAU.1				X		
FIA_SOS.1	X				X	
FPT_FLS.1	X		X			
FPT_ITC.1		X				
FPT_ITI.1		X				
FPT_TEE.1	X	X	X			

7.3.2 SFR Rationale

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

OT.PREVENT_BF is provided by the following SFRs:

- FCS_COP.1/RSA_SCD, FCS_COP.1/RSA_Other, FCS_COP.1/AES, FCS_COP.1/HMAC and FCS_COP.1/SHA-2 ensure that well-known secure cryptographic algorithms are used and the signature algorithms are robust.
- FIA_SOS.1 ensures that D.PIN chosen by the Signer is not easily guessed.
- FPT_FLS.1 along with the FPT_TEE.1 ensures that the random number generator provided by the mobile operating systems is with good quality and produces cryptographically strong keys.

OT.SECURE_CHANNEL is provided by the following SFRs:

- FCS_COP.1/RSA_SCD, FCS_COP.1/RSA_Other, FCS_COP.1/AES, FCS_COP.1/HMAC and FCS_COP.1/SHA-2 ensure that well-known secure cryptographic algorithms are used for channel encryption and the key wrapping.

- FPT_ITC.1 along with the FPT_ITI.1 ensures the confidentiality and the integrity of the components of the SCD (D.serverPart) and components of the signature (D.application SignatureShare), when transmitted to the Smart-ID SecureZone.
- FPT_TEE.1 ensures that the connection with the Smart-ID Core is positively authenticated.
- FCS_CKM.1/DH_TEK ensures that TOE and SecureZone securely establish the encryption key for communication.

OT.SECURE_CRYPTO is provided by the following SFRs:

- FCS_CKM.1/RSA_clientShare and FCS_CKM.1/DH_TEK ensure that secure cryptographic algorithms are used to generate the SVD.
- FCS_CKM.4 ensures that components of the SCD, which are generated inside the TOE (D.clientShare, D.clientPart, D.serverPart) are securely destroyed, when no longer required.
- FCS_COP.1/RSA_SCD, FCS_COP.1/RSA_Other, FCS_COP.1/AES, FCS_COP.1/HMAC and FCS_COP.1/SHA-2 ensure that well-known secure cryptographic algorithms are used and the signature algorithms are robust.
- FPT_FLS.1 ensures that TOE fails securely and preserves the secure state.
- FPT_TEE.1 ensures that the random number generator provided by the mobile operating systems is with good quality and produces cryptographically strong keys.

OT.VERIFICATION_CODE is provided by the FDP_DAU.1, which ensures that TOE computes the verification code, which represents the context of the authentication/signing session and allows the U.User to verify that the D.DTBS/R is the one, which is intended to be signed by him or her.

OT.ENC_STORAGE is provided by the following SFRs:

- FCS_COP.1/RSA_SCD, FCS_COP.1/RSA_Other, FCS_COP.1/AES, FCS_COP.1/HMAC and FCS_COP.1/SHA-2 ensure that well-known secure cryptographic algorithms are used for the storage encryption.
- FIA_SOS.1 ensures that D.PIN chosen by the Signer is not easily guessed.

OT.CLONE_DETECTION is provided by the FPT_CLD.1, which ensures that TOE follows the clone detection protocol with the API call and performs the periodic updates of the D.OTP data.

7.3.3 SFR Dependencies Analysis

Table 11 shows how the dependencies of the SFRs is fulfilled.

Table 11. Analysis of fulfillment of SFR dependencies

SFR	Dependecies	Fulfilled by
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/*

Table 11. Analysis of fulfillment of SFR dependencies

SFR	Dependecies	Fulfilled by
	FCS_CKM.4	FCS_CKM.4
FPT_CLD.1	none	none
FDP_DAU.1	none	none
FIA_SOS.1	none	none
FPT_FLS.1	none	none
FPT_ITC.1	none	none
FPT_ITI.1	none	none
FPT_TEE.1	none	none

7.4 Security Assurance Requirements

7.4.1 Rationale for selecting the SARs

The assurance level for this ST is chosen to be the EAL2. EAL2 is usually applicable in the situations, where users require moderate level of independently assured security, without additional effort from the developer side, other than is consistent with good commercial practice. EAL2 is the lowest level, which includes the vulnerability analysis with the penetration testing, which gives the assurance that the TOE is resistant to attackers.

As such, EAL2 is appropriate for the software library, which is to be embedded in the app of the mobile device.

7.4.2 Security assurance components

The security assurance requirements are drawn from CC and represent the EAL2. The assurance components are identified in the table 12.

Table 12. 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.2) Basic modular design (ADV_TDS.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.2)

	Problem tracking CM coverage (ALC_CMS.2) Delivery procedures (ALC_DEL.1)
Tests (ATE)	Functional testing (ATE_FUN.1) Analysis of coverage (ATE_COV.1) Independent testing - sample (ATE_IND.2)
Vulnerability Assessment	Vulnerability analysis (AVA_VAN.2)

7.4.3 SAR dependencies analysis

The security assurance requirements are drawn from CC and represent the standard EAL2 assurance package. As such, all the dependencies of the individual assurance components are already met.

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 7.2 – 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.

8.1 Trusted Channels

8.1.1 SF.SecureChannel

The TOE uses the D.KTK and D.TEK to encrypt communication, the security of the data flow, between the TOE and the Smart-ID backend components is protected by following mechanisms:

1. HTTPS connections, which provide the confidentiality and detection of the modifications
2. HTTPS pinning, which authenticate the Smart-ID backend component
3. HTTP Basic Auth, which authenticates the TOE to the Smart-ID backend
4. JWE encryption, which ensures that certain data fields in the messages can only be read by the Smart-ID SecureZone component.

This SF implements the FCS_CKM.1/DH_TEK, FCS_COP.1/RSA_SCD, FCS_COP.1/RSA_Other, FCS_COP.1/AES, FCS_COP.1/HMAC, FCS_COP.1/SHA-2, FPT_ITC.1, FPT_ITI.1 and FPT_TEE.1.

8.2 Handling of cryptographic material and algorithms

8.2.1 SF.CryptoAlgorithms - Using standard cryptographic algorithms

The TOE uses the following cryptographic algorithms that meet or exceed the requirements of ETSI and this ensures that signatures cannot be forged:

- RSAES-PKCS1-v1_5 signature scheme with keys from 2048-bits up to 16384 bits; (ETSI recommended minimum 2048-bits)
- RSAES-OAEP decryption scheme with keys from 2048-bits up to 16384 bits; (ETSI recommended minimum 2048-bits)
- SHA-256 hash algorithm
- AES encryption algorithm with 128-bit or 256-bit keys.

Applicable standards:

- FIPS-186-4 - “Digital Signature Standard”.
- RFC8017 - “PKCS #1: RSA Cryptography Specifications Version 2.2 ”.
- NIST SP 800-131Ar1 - “Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths”.
- NIST SP 800-57 Part 1 Revision 4 - “Recommendation for Key Management. Part 1: General”.

This SF implements the FCS_CKM.1/RSA_clientShare, FCS_CKM.1/DH_TEK, FCS_COP.1/AES, FCS_COP.1/RSA_SCD, FCS_COP.1/RSA_Other, FCS_COP.1/HMAC and FCS_COP.1/SHA-2.

8.2.2 SF.KeyGen - Key generation and registration

The TOE generates the keypair (D.clientShare and D.clientModulus) with the RSA key generation algorithm. The TOE splits the D.clientShare into two parts (D.clientPart and D.serverPart) and then transmits the D.serverPart along with the D.clientModulus to the Smart-ID SecureZone with the key distribution method (encryption with the key transport key).

This SF implements the FCS_CKM.1/RSA_clientShare, FCS_COP.1/RSA_SCD and FCS_CKM.1/DH_TEK.

8.2.3 SF.EncryptedStorage - Secure data storage

The TOE securely stores the cryptographic key material outside of the TOE, within the app’s sandbox storage area. The key material is encrypted with the following methods:

1. AES encryption algorithm.

This SF implements the FCS_COP.1/AES and FCS_COP.1/HMAC.

8.2.4 SF.Signing - Generating the signature share

The TOE receives the D.DTBS/R from the SCA and computes the verification code from the DTBS/R. The verification code is displayed to the U.User. U.User confirms that he/she wishes to sign the D.DTBS/R associated with the verification code and enters the D.PIN. The TOE gets the status information from SecureZone about the key pair. The TOE decrypts the D.clientPart and computes the D.applicationSignaturePart. The D.applicationSignaturePart is then encrypted with the key transport key and transmitted to the Smart-ID SecureZone.

This SF implements the FCS_CKM.1/RSA_clientShare, FCS_COP.1/RSA_SCD, FCS_COP.1/AES, FCS_COP.1/HMAC, FDP_DAU.1, FPT_ITC.1 and FPT_ITI.1.

8.2.5 SF.KeyZer - Key destruction

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

1. D.clientShare,
2. D.clientPart,
3. D.serverPart.

The keys are deleted by overwriting the content of the key storage blob object in the storage.

This SF implements the FCS_CKM.4.

8.3 Signatory authentication data

8.3.1 SF.CloneDetection

The TOE uses D.OTP for performing subsequent API Call, and also follows the Clone Detection protocol supplies new one-time-password after each API call. This requires management of the state machine inside the TOE. Also, TOE sends periodic queries (refreshCloneDetection) to the Smart-ID backend, to receive updates to the D.OTP.

This SF implements the FPT_CLD.1

8.3.2 SF.PINQuality

The TOE verifies the user-supplied D.PIN against the blacklist and against the length limitation. In case the D.PIN is too simple or too short, the user registration is denied.

This SF implements the FIA_SOS.1

8.4 Failure modes and reliability

8.4.1 SF.SecurityTest - Testing external entities

The TOE performs the following tests to verify that external entities are correct:

- consistency of the configuration data,
- statistical quality of the environment provided PRNG,
- positive authentication of the Smart-ID backend services with the HTTPS pinning.

The statistical quality of the environment provided PRNG is verified by implementing the following tests from the suite SP 800-22r1a [9]:

1. monobit test
2. poker test
3. runs test
4. long-runs test

In case the TOE fails with the testing, the TOE aborts the key generation process or the connection to the Smart-ID backend services.

This SF implements the FPT_TEE.1 and FPT_FLS.1.

8.5 TOE Summary Specification Rationale

The table 13 shows the mapping between SFRs and TOE Security Functions to provide the quick overview.

Table 13. Mapping between SFRs and TSF

SFR	SF
FCS_CKM.1/RSA_ clientShare	SF.CryptoAlgorithms SF.Signing SF.KeyGen
FCS_CKM.1/DH_TEK	SF.CryptoAlgorithms SF.SecureChannel SF.KeyGen
FCS_CKM.4	SF.KeyZer
FCS_COP.1/RSA_SCD	SF.SecureChannel SF.CryptoAlgorithms SF.KeyGen SF.Signing
FCS_COP.1/RSA_Other	SF.SecureChannel SF.CryptoAlgorithms
FCS_COP.1/AES	SF.SecureChannel SF.CryptoAlgorithms SF.Signing SF.EncryptedStorage
FCS_COP.1/HMAC	SF.SecureChannel SF.CryptoAlgorithms SF.Signing SF.EncryptedStorage
FCS_COP.1/SHA-2	SF.SecureChannel SF.CryptoAlgorithms
FPT_CLD.1	SF.CloneDetection
FDP_DAU.1	SF.Signing
FIA_SOS.1	SF.PINQuality
FPT_FLS.1	SF.SecurityTest
FPT_TEE.1	SF.SecurityTest SF.SecureChannel
FIA_ITC.1	SF.SecureChannel SF.Signing
FIA_ITI.1	SF.SecureChannel SF.Signing