STeID JC Open OS v1.0
Security Target Lite

Version: B
Date: 2025-10-24
STMicroelectronics

Document history

Version	Date	Comment	Author
A	October 06, 2025	First release	STMicroelectronics
B	October 24, 2025	Second release	STMicroelectronics

Distribution list

The document is public.

Contents

1 ST Introduction

This section provides information about the TOE, which enables a potential user of the TOE to determine, whether the TOE implements the functionality required by the user.

1.1 ST Reference

Title	STeID JC Open OS v1.0 Security Target Lite
Version	See Document history
Date	See Document history
Author	STMicroelectronics

Table 1 Security Target reference

1.2 TOE Reference

TOE Name	SteID JC Open OS v1.0	
TOE Version	1.3.2	
TOE Identification	**IC**	IC Name: ST31N600 IC Maskset name: K470B Version: A03 Master product identification number: 0x0200 Firmware version: 3.1.3
	Java Card OS	OS_IDENTIFIER: 0x0900 OS_RELEASE_DATE: 0x5246 OS_RELEASE_LEVEL: 0x0002 OS_VERSION: 00010302
TOE Type	Embedded secure element (eSE) with a Java Card System	

Table 2 TOE reference

1.2.1 Other certifications

The ST31N600 Secure IC has been already certified:

1. IC name: ST31N600 A03
2. CC certificate reference [CERT-IC].

1.3 **TOE Overview**

SteID JC Open OS v1.0 (in this document also referenced to as the TOE) system-on-chip is an embedded secure element (eSE) with a Java Card System compliant with Java Card specifications version 3.0.5 with all the mandatory features, plus the following additions:

1. support for the *int* type (including the *intx* package) and object deletion.
2. support for Sensitive Results augmentation package.

The TOE can host and manage Java Card applets from different stakeholders (user, original equipment manufacturer (OEM), hardware integrator, service provider).

The TOE offers the following capabilities:

3. Java Card™ open platform, version 3.0.5 Classic Edition, the product includes a fully functional Java Card Virtual Machine [JCVM], a Java Card Runtime Environment [JCRE] and Java Card API [JCAPI] compliant to Java Card 3.0.5 specification.
4. GlobalPlatform® version 2.3.1
    1. Amendment D
    2. Amendment H
5. Cryptographic functions (some cryptographic algorithms can be disabled during personalization)
6. Proprietary API for Secure Messaging (encryption and MAC computation using DES and AES algorithms as outlined in [ICAO] 9303, Part 11 for Machine Readable Travel Documents). This feature can be disabled during personalization.
7. Communication ISO 7816-3 (T=0, T=1), ISO 14443 up to 848 kbps, Dual-interface support
8. Physical Protection against physical tampering and leakage;
9. **PIN**. The TOE implements secure PIN compare functions and PIN integrity protection;
10. **Firmware upgrade**. The product relies on IC's certified secure loader (i.e. "Loader dedicated for usage by authorized users only", from now one mentioned as "Secure Loader") to patch the operational OS firmware at OEM factory or on the field. Firmware upgrade can be disabled during personalization.
11. **Biometric API (**this feature can be disabled during personalization)**.**

Other features included in the TOE but out of TSF are:

12. Match-on-Card library (fingerprints and face functionality).

1.3.1 Non-TOE Hardware/Software/Firmware

Here is a description of the non-TOE components and systems:

Component	**Required**	**Description**

Bytecode verifier	Mandatory	The bytecode verifier is a program that performs static checks on the bytecodes of the methods of a CAP file prior to the execution of the file on the card. Bytecode verification is a key component of security: applet isolation, for instance, depends on the file satisfying the properties a verifier checks to hold. A method of a CAP file that has been verified shall not contain, for instance, an instruction that allows forging a memory address or an instruction that makes improper use of a return address as if it were an object reference. In other words, bytecodes are verified to hold up to the intended use to which they are defined. Bytecode verification could be performed totally or partially dynamically. No standard procedure in that concern has yet been recognized. Furthermore, different approaches have been proposed for the implementation of bytecode verifiers, most notably data flow analysis, model checking and lightweight bytecode verification, this latter being an instance of what is known as proof carrying code. The actual set of checks performed by the verifier is implementation-dependent, but it is required that it should at least enforce all the "must clauses" imposed in [JCVM] on the bytecodes and the correctness of the CAP files' format.

Table 3 Components of the environment

1.4 **TOE Description**

The TOE is a composition of a Java Card OS in open configuration and crypto library with the ST31N600 IC platform.

It is designed to host and manage Java Card applets from different stakeholders.

TOE usage is focused on security critical applications in small form factors, main usage scenarios are: electronic passports, secure elements used for device authentication (where the TOE can be used to prove the authenticity or originality of a device), banking cards, electronic drivers' licenses, eSignature, eVoting, Online Authentication.

Figure 1 shows the high-level architecture of the TOE, in the pink box the TOE perimeter is delimited, software components and interfaces out of the pink box are out of scope. Match on Card library is dashed because part of the TOE but not part of the TSF.

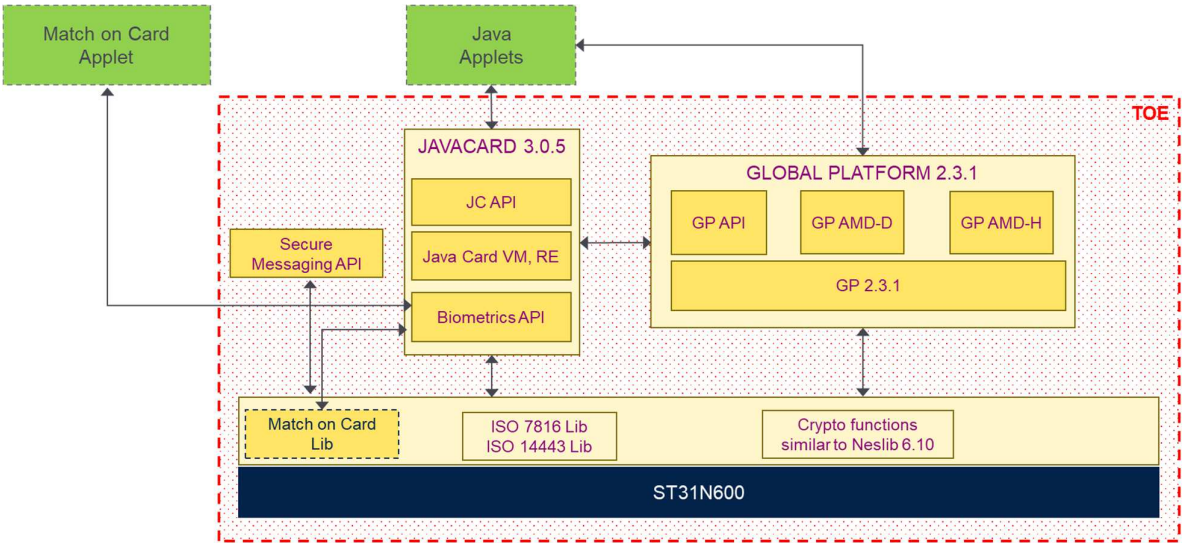

Figure 1 TOE Components

Hardware description

The hardware is the ST31N600 secure microcontroller, specific versions of the hardware parts are described in Section 1.2.

Java Card

Java Card functionality is included in the TOE; more specifically, the product includes a fully functional Java Card Virtual Machine [JCVM], a Java Card Runtime Environment [JCRE] and Java Card API [JCAPI] compliant to Java Card 3.0.5 specification.

1.4.1 Physical Scope

The TOE is a composite TOE comprising hardware and software. The physical scope is defined as:

1. the STMicroelectronics IC ST31N600 Security Integrated Circuit with dedicated software Common Criteria certified by NSCIB with assurance level EAL6+ [CERT-IC].
2. An encrypted image of the TOE Operating system (TOE OS), including:
    1. the Java Card Operating System version 3.0.5.
    2. cryptographic library.
    3. system applications with their configuration data.
3. the associated guidance documentation delivered in .pdf format delivered encrypted by e-mail:
    1. Operational User Guidance [AGD_OPE]
    2. Preparative Procedure [AGD_PRE]
4. documentation of the proprietary SecureMessaging API, delivered on request by means of customer support in zip format and encrypted by pgp:
    1. SecureMessaging JavaDoc Rev. B

The encrypted image of the TOE OS is transferred to STMicroelectronics engineering department encrypted via PGP by using shared repositories.

The TOE will be delivered by a trusted courier at the end of the phase B (see Section 1.4.4) either as sawn wafers or micromodules, or contactless modules (with/without inlay), with OS, personalization keys and data preloaded, in operational mode.

1.4.2 Logical Scope

The main security functions of the TOE are:

1. **Firewall**. The TOE implements an applet firewall according to [JCRE].

2. **Sensitive data confidentiality**. The TOE ensures that sensitive information is made unavailable after deletion.

3. **Rollback protection**. The TOE implements atomicity and rollback mechanism for Java Card runtime environment [JCRE]

4. **Secure Communications**. The TOE implements secure channel protocols according to [GP], chapter 10 and GP Amendment D[GP-D].

5. **Card Management**. The TOE supports the GlobalPlatform Card Specifications v.2.3 and related amendments:

    1. GlobalPlatform Amendment D – Secure Channel Protocol SCP03 v1.1.1
    2. GlobalPlatform Amendment H – Executable Load File Upgrade v1.1

    A Card Manager (Issuer Security Domain) is present on the product.

6. **Physical Protection**. The TOE provides means to protect itself against physical tampering and leakage.

7. **Cryptographic Support**. The TOE provides key creation, key management, key deletion and cryptographic functionality.

    It provides Secure Messaging API to offer encryption and MAC computation services needed by ICAO applications as specified in [ICAO].

    It also provides the API in accordance to the Java Card API Specification [JCAPI]. In the following table the options supported by this TOE are generically listed; please refer to FCS_COP SFR for the exhaustive list of algorithms, modes, padding schemes:

Algorithm Family	**Cryptographic functionalities**	**Cryptographic key size (in bits)**
AES	AES key generation; Encryption, Decryption; MAC computation	128, 192, 256
TDES	TDES key generation; Encryption, Decryption; MAC computation;	112, 168
Diffie Hellman	DH key pair generation	2048
RSA	RSA key pair generation (private CRT and public); Encryption, Decryption; Signature generation and verification;	CRT up to 4096 bits
ECC	EC key pair generation; ECDH key agreement; ECDSA Signature generation and verification	160, 192, 224, 256, 320, 384, 512, 521
HMAC	Signature generation and verification	N.A.
HASH	SHA-1, SHA-224, SHA-256, SHA-384, SHA-512; Protected SHA-1, Protected SHA-256, Protected SHA-384, Protected SHA-512	NA
ICAO Secure Messaging	Encryption and MAC generation; MAC verification and decryption	AES: 128, 192, 256 DES: 112, 168

8. **Generation of Random Numbers.** Secure random number generation mechanisms compliant to PTG.2 Class and DRG.3 Class.

9. **PIN**. The TOE implements secure PIN compare functions and PIN integrity protection;

10. **Firmware upgrade**. The product also relies on IC's certified secure loader (i.e. "Loader dedicated for usage by authorized users only", from now one mentioned as "Secure Loader") to patch the operational OS firmware at OEM factory or on the field.

The following listed features are optional removable features:

- HMAC JC API support
- AEAD ciphers JC API support
- RSA Key generation
- Firmware upgrade enabler / OS upgrade
- Biometry official API and provider implementation
- ICAO secure messaging

1.4.3 TOE composition and identification

The TOE is the composition of a Java Card OS and the cryptolibrary on top of ST31N600 IC. The chip has been certified Common Criteria.

As defined in [ADG_OPE], as the TOE is made by two Operating systems (Operational Java Card OS and Firmware Loader certified with the IC), the two OS-es have independent identification information that can be retrieved by issuing a GET DATA command with two dedicated tags to the Issuer Security Domain available on the Java Card OS..

Both GET DATA tags allow also to identify the Hardware.

The TOE certification applies to the versions defined in Section 1.2

For further details, refer to [AGD_OPE].

1.4.4 TOE Life Cycle

The composite product life cycle is decomposed into 4 phases. Each of these phases has the very same boundaries as those defined in the claimed protection profile [PP-JC].

The life cycle phases are summarized in Table 4.

Phase	Name	Description
A	JCS Development	This phase corresponds to the first two stages of the IC development. In this phase the OS and related applications are developed according to the Phase 1 of the ST Life cycle model as reported in Operational User Guidance. JCS encrypted image is delivered at the end of this phase.
B	JCS Storage, Pre-personalization testing	This phase corresponds to phase 5 of the IC development. In this phase the encrypted image is downloaded on the hardware by using the Flash Loader according to IC procedures. Product configuration is performed, including all the applications integration, the system applications configurations and static data configuration, according to Phase 5 of ST life cycle model reported in Operational User Guidance. TOE Delivery happens at the end of this phase.

C	JCS Personalization	This phase corresponds to phase 6 of the IC development. In this phase, the devices are personalized with diversified credentials, according to Phase 6 of ST life cycle model reported in Operational User Guidance.
D	JCS Final usage	This phase corresponds to phase 7 of the IC development. Such a phase represents the life cycle state of the product on the field, according to Phase 7 of ST life cycle model reported in Operational User Guidance.

Table 4 TOE life cycle phases

2 Conformance claims

2.1 CC Conformance Claims

The TOE and ST claim conformance to the CC Version 3.1 revision 5 [CC31R5P2] [CC31R5P3].

The ST claim conformance to CC Part 2 extended and CC Part 3 conformant.

2.2 Package Claims

ST claims conformance to assurance package EAL6 augmented with ALC_FLR.2.

2.3 PP Claims

ST claims demonstrable conformance to:

1. Java Card Protection Profile – Open Configuration Version 3.1 PP-JC.

This ST is more restrictive than the PP [PP-JC] and Section 9.1 “Conformance Claim Rationale” provides a rational for it.

2.4 Conformance Rationale

This Security Target claims demonstrable conformance to the protection profile PP-JC.

The Security Assurance Requirements statement for the TOE in this Security Target includes all the requirements for the TOE from the PP-JC.

2.4.1 Security Problem Definition Statement

All sections of this Security Target regarding the Security Problem Definition, Security Objectives Statement and Security Requirements Statement for the TOE are taken over from PP-JC with the exception described in the following table.

SPD from [PP-JC]	Description
A.DELETION	Deletion of applets is in the scope of the evaluation. As discussed in Section2.4.2, O.CARD_MANAGEMENT is now Security Objective for the TOE.

Table 5 Security Problem Definition Statement

In addition, the Sensitive Result augmentation packages from [PP-JC] is in the scope. The SPD of this optional package is taken from Appendix 2 of the Java Card PP [PP-JC].

Additional threats and OSPs have been defined for Firmware Upgrade in section 3.2.

2.4.2 Security Objectives Statement

The Security Objectives for the TOE and the Operational Environment of the Java Card implementation are the same as in the Java Card PP PP-JC with the exceptions described in the following table.

SO from [PP-JC]	Description
O.ARRAY_VIEWS_CONFID	Functionality not implemented and therefore SO is not claimed.
O.ARRAY_VIEWS_INTEG	Functionality not implemented and therefore SO is not claimed.

OE.CARD-MANAGEMENT	Card Manager is part of the TOE. Replaced by O.CARD-MANAGEMENT.
OE.SCP.RECOVERY	Request on the Security IC component. Replaced by O.SCP.RECOVERY.
OE.SCP.SUPPORT	Request on the Security IC component. Replaced by O.SCP.SUPPORT.
OE.SCP.IC	Request on the Security IC component. Replaced by O.SCP.IC.

Table 6 Java Card security objective statement

For the Java Card functionality, the Sensitive Result augmentation package from [PP-JC] is in the scope. Security Problem Definition of this optional package is taken from Appendix 2 of Java Card PP [PP-JC].

Additional Security Objectives have been defined for Firmware Upgrade in section 4.2.

2.4.3 Security Functional Requirements Statement

2.4.3.1 Java Card

The Security Functional Requirements for the Java Card component are taken from the Java Card PP PP-JC without any modification. The Java Card OS also implements SFRs from the augmentation package Sensitive Results according to the Java Card PP Appendix 2 PP-JC.

Additional SFRs have been defined for Firmware Upgrade in section 6.2.

3 Security Problem Definition

3.1 Java Card

The Security Problem Definition for the Java Card implementation is the same as the Security Problem Definition described in the Java Card PP PP-JC with the exceptions described in Section 2.4.1.

The TOE implements the following augmentation packages defined in Appendix 2 of the Java Card PP PP-JC: Sensitive Result.

The Security Problem Definition for this augmentation package is taken from the Java Card PP PP-JC.

3.2 Firmware Upgrade OS

Application note: Firmware upgrade can be disabled during personalization. In case it is disabled the following SPD do not apply.

3.2.1 Assets

Assets	Description
Software image	The software image running on the TOE, which can be updated by a valid authenticated user through the Secure Loader functionality.

Table 7 Assets

3.2.2 Subjects

Subjects	Description
Loader user	User accessing the Secure Loader functionality to perform a software loading operation.

Table 8 Subjects

3.2.3 Threats

Threats	Description
T.LOADER_MISUSE	An attacker performs unauthorised use of the software loader functionality to upload a modified or malicious software version.

Table 9 Threats

3.2.4 Organisational Security Policies

Policies	Description
P.KEY_PERSO	Operational OS credentials (see Content loading key and Firmware Upgrade Authorization keys definitions in [AGD_OPE], section 7.4)are updated during the JCS Personalization phase of the TOE. After all operational OS keys are set, the TOE state is changed to JCS Final usage phase.

Table 10 Organizational Security Policies

4 Security Objectives

4.1 **Java Card**

4.1.1 Security Objectives for the TOE

The Security Objectives for the TOE for the Java Card implementation are taken from the Security Objectives for the TOE described in the Java Card PP PP-JC with the exceptions and precisions described in Section 2.4.2.

The TOE implements the following augmentation packages defined in Appendix 2 of the Java Card PP PP-JC: Sensitive Result. The Security Objectives for the TOE added by the augmentation packages are taken from the Java Card PP PP-JC.

As also anticipated in Section 2.4.2, some of the Security Objectives for the Operational Environment from PP-JC are listed as TOE Security Objectives in this ST: O.SCP.RECOVERY instead of OE.SCP.RECOVERY O.SCP.SUPPORT instead of OE.SCP.SUPPORT, O.SCP.IC instead of OE.SCP.IC and O.CARD-MANAGEMENT instead of OE.CARD-MANAGEMENT. They are detailed in the following table:

SO for the TOE	Description
O.CARD-MANAGEMENT	The card manager shall control the access to card management functions such as the installation, update or deletion of applets, as well as GP registry updates. It shall also implement the card issuer's policy on the card. The card manager is an application with specific rights, which is responsible for the administration of the smart card, it is in charge of the life cycle of the whole card, as well as that of the installed applications (applets). The card manager should prevent that card content management (loading, installation, deletion) is carried out, for instance, at invalid states of the card or by non-authorized actors. It shall also enforce security policies established by the card issuer.
O.SCP.IC	The SCP shall provide all IC security features against physical attacks. This security objective refers to the point (7) of the security aspect #.SCP: It is required that the IC is designed in accordance with a well-defined set of policies and Standards (likely specified in another protection profile), and will be tamper resistant to actually prevent an attacker from extracting or altering security data (like cryptographic keys) by using commonly employed techniques (physical probing and sophisticated analysis of the chip). This especially matters to the management (storage and operation) of cryptographic keys.
O.SCP.RECOVERY	If there is a loss of power, or if the smart card is withdrawn from the CAD while an operation is in progress, the SCP must allow the TOE to eventually complete the interrupted operation successfully, or recover to a consistent and secure state. This security objective refers to the security aspect #.SCP(1): The smart card platform must be secure with respect to the SFRs. Then after a power loss or sudden card removal prior to completion of some communication protocol, the SCP will allow the TOE on the next power up to either complete the interrupted operation or revert to a secure state.
O.SCP.SUPPORT	The SCP shall support the TSFs of the TOE. This security objective refers to the security aspects 2, 3, 4 and 5 of #.SCP: • (2) It does not allow the TSFs to be bypassed or altered and does not allow access to other low-level functions than those made available by the packages of the API. That includes the protection of its private data and code (against disclosure or modification) from the Java Card System. • (3) It provides secure low-level cryptographic processing to the Java Card System. • (4) It supports the needs for any update to a single persistent object or class field to be atomic, and possibly a low-level transaction mechanism. (5) It allows the Java Card System to store data in "persistent technology memory" or in volatile memory, depending on its needs (for instance, transient objects must not be stored in non-volatile memory). The memory model is structured and allows for low-level control accesses (segmentation fault detection).

Table 11 Additional Security Objectives for the TOE

4.1.2 Security Objectives for the Operational Environment

The Security Objectives for the Operational Environment for the Java Card implementation are taken from the Security Objectives for the Operational Environment described in the Java Card PP with the exceptions discussed in Section 2.4.2.

4.1.3 Security Objectives Rationale

The Security Objectives Rationale for the Java Card implementation are taken from the Security Objectives Rationale section described in the Java Card PP PP-JC with the exceptions discussed in Sections 2.4.1 and 2.4.2 and justified below:

1. assumption A.Deletion is excluded. Being the Card Manager part of the TOE, the assumption is no longer relevant. Leaving out the assumption, makes the SPD of this ST more restrictive than the SPD in the JCPP. As the Card Manager is part of the TOE, it is ensuring that the deletion of applets through the Card Manager is secure, instead of assuming that it is handled by the Card Manager in the environment of the TOE.
2. O.SCP.RECOVERY, O.SCP.SUPPORT, and O.SCP.IC are objectives for the TOE as the Smart Card Platform belongs to the TOE for this evaluation. O.CARDMANAGEMENT is an objective for the TOE as the Card Manager belongs to the TOE for this evaluation. Moving objectives from the environment to the TOE adds objectives to the TOE without changing the overall objectives. The statement of security objectives is therefore equivalent to the security objectives in the Java Card Protection Profile to which conformance is claimed.

The TOE implements the following augmentation packages defined in Appendix 2 of the Java Card PP PP-JC: Sensitive Result. The Security Objectives Rationale added by the augmentation packages are taken from the Java Card PP PP-JC.

4.2 **Firmware Upgrade Functionality**

Application note: Firmware upgrade can be disabled during personalization. In case it is disabled the following Security Objectives do not apply.

4.2.1 Security Objectives for the TOE

Objectives	**Description**
OT.ACCESS_CONTROL	The TOE shall provide access control mechanisms to ensure only valid authenticated users can access the TOE functionality, i.e. Secure Loader functionality.

Table 12 Security Objectives for the TOE

4.2.2 Security Objectives for the Operational Environment

Objectives	Description
OE.KEY_PERSO	The operational environment shall ensure that when the TOE life cycle is in manufacturing state (CMS), and before it is set to release state (URS), all the default keys in the TOE are updated with final usage phase keys, FW authentication keys and the content loading keys.

Table 13 Security Objectives for the Operational Environment

4.2.3 Security Objectives rationale

The following table shows how the security objectives for the TOE cover the threats of the Firmware Upgrade OS functionalities.

Threats / Security Objectives	**OT.ACCESS_CONTROL**
T.LOADER_MISUSE	X

Table 14 Mapping of threats to TOE security objectives

T.LOADER_MISUSE is covered by the security objective OT.ACCESS_CONTROL, which ensures that all the TOE functionality (Loader functionality) can only be accessed by valid authenticated users.

The following table shows how the security objectives for the operational environment cover the OSP.

OSPs / Security Objectives	**OE.KEY_PERSO**
P.KEY_PERSO	X

Table 15 Mapping of OSP to security objectives of the environment

The OSP P.KEY_PERSO is covered by the environment security objective OE.KEY_PERSO, which enforces that the default TOE keys are updated by the manufacturer before the TOE is set to the final usage phase (URS).

5 Extended Component Definition

5.1 Java Card

Extended Component Definition from the Java Card PP-JC has been taken with no modification.

6 Security Functional Requirements

Reading notes:

(1) Selections having been made by the PP author are denoted as underlined text.
(2) Selections filled in by the ST author appear in square brackets with an indication that a selection is to be made [selection:] and are italicised.
(3) Assignments having been made by the PP author are denoted by showing as bold text.
(4) Assignments filled in by the ST author appear in square brackets with an indication that an assignment is to be made [assignment:] and are italicised.
(5) Refinements, if applicable, have been identified in bold and italicised text.
(6) Iteration is denoted by showing a slash "/", and the iteration indicator after the component identifier.

6.1 **Java Card**

6.1.1 COREG_LC SECURITY FUNCTIONAL REQUIREMENTS

The following table shows all the SFRs from Java Card PP PP-JC that do not require to perform any operation and therefore are an exact copy of the PP. SFRs needing refinement, assignment or selection operations with respect to Java Card PP PP-JC definitions are addressed in the next sections.

Section	**SFR**
Firewall Policy	FDP_ACC.2/FIREWALL Complete access control
	FDP_ACF.1/FIREWALL Security attribute based access control
	FDP_IFC.1/JCVM Subset information flow control
	FDP_RIP.1/OBJECTS Subset residual information protection
	FMT_MSA.1/JCRE Management of security attributes
	FMT_MSA.1/JCVM Management of security attribute
	FMT_MSA.2/FIREWALL_JCVM Secure security attributes
	FMT_MSA.3/FIREWALL Static attribute initialisation
	FMT_MSA.3/JCVM Static attribute initialisation
	FMT_SMF.1 Specification of Management Functions
	FMT_SMR.1 Security roles
Application Programming Interface	FDP_RIP.1/ABORT Subset residual information protection
	FDP_RIP.1/APDU Subset residual information protection
	FDP_RIP.1/GlobalArray Subset residual information protection
	FDP_RIP.1/bArray Subset residual information protection
	FDP_RIP.1/KEYS Subset residual information protection
	FDP_RIP.1/TRANSIENT Subset residual information protection
	FDP_ROL.1/FIREWALL Basic rollback
Card Security Management	FPT_FLS.1 Failure with preservation of secure state
AID Management	FIA_ATD.1/AID User attribute definition
	FIA_UID.2/AID User identification before any action
	FMT_MTD.1/JCRE Management of TSF data

	FMT_MTD.3/JCRE Secure TSF data

6.1.1.1 Firewall policy

6.1.1.1.1 FDP_IFF.1/JCVM Simple security attributes

FDP_IFF.1.1/JCVM The TSF shall enforce the **JCVM information flow control SFP** based on the following types of subject and information security attributes:

Subjects	**Security attributes**
S.JCVM	Currently Active Context

FDP_IFF.1.2/JCVM The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold:

1. **An operation OP.PUT(S1, S.MEMBER, I.DATA) is allowed if and only if the Currently Active Context is "Java Card RE";**
2. **o other OP.PUT operations are allowed regardless of the Currently Active Context's value.**

FDP_IFF.1.3/JCVM The TSF shall enforce the [assignment: *no additional control SFP rules*].

FDP_IFF.1.4/JCVM The TSF shall explicitly authorise an information flow based on the following rules: [assignment: *none*].

FDP_IFF.1.5/JCVM The TSF shall explicitly deny an information flow based on the following rules: [assignment: *none*].

Application Note:
The storage of temporary Java Card RE-owned objects references is runtime-enforced ([JCRE], §6.2.8.1-3).

It should be noticed that this policy essentially applies to the execution of bytecode. Native methods, the Java Card RE itself and possibly some API methods can be granted specific rights or limitations through the FDP_IFF.1.3/JCVM to FDP_IFF.1.5/JCVM elements. The way the Java Card virtual machine manages the transfer of values on the stack and local variables (returned values, uncaught exceptions) from and to internal registers is implementation-dependent. For instance, a returned reference, depending on the implementation of the stack frame, may transit through an internal register prior to being pushed on the stack of the invoker. The returned bytecode would cause more than one OP.PUT operation under this scheme.

6.1.1.2 Application Programming Interface

6.1.1.2.1 FCS_CKM.1 Cryptographic key generation

FCS_CKM.1.1 The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm [assignment: *see table below*] and specified cryptographic key sizes [assignment: *see table below*] that meet the following: [assignment: *see table below*].

Application Note:

1. The keys can be generated and diversified in accordance with [JCAPI] specification in classes KeyBuilder and KeyPair (at least Session key generation).
2. This component shall be instantiated according to the version of the Java Card API applying to the security target and the implemented algorithms ([JCAPI]).

Refer to Appendix 4 PP-JC to define the allowed/available key generation algorithms as per Java Card API specifications [JCAPI], in the table below the options supported by this TOE are reported[1].

Iteration	**Cryptographic key generation algorithm**	**Cryptographic key size (in bits)**	**List of standards**
AES	AES key generation	128, 192, 256	FIPS PUB 197
TDES	TDES key generation	112, 168	FIPS PUB 46-3 (ANSI X3.92) FIPS PUB 81 GlobalPlatform v2.3
Diffie Hellman key generation	DH key pair generation	(2048, 224) (2048, 256)	ANSI X9.42
RSA key generation	RSA key pair (private CRT and public)	CRT up to 4096 bits	ISO/IEC 9796-2 PKCS#1 v2.1
ECC	EC key pair generation	160, 192, 224, 256, 320, 384, 512, 521 bits	[ANSI X9.62] [ISO 14888-3] [FIPS 186-4]

Application Note: some cryptographic algorithms can be disabled during personalization.

6.1.1.2.2 FCS_CKM.4 Cryptographic key destruction

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method [assignment: *overwriting the keys with zeros*] that meets the following: [assignment: *none*].

Application Note:

1. The keys are reset as specified in [JCAPI] Key class, with the method clearKey(). Any access to a cleared key for ciphering or signing shall throw an exception.
2. This component shall be instantiated according to the version of the Java Card API applicable to the security target and the implemented algorithms [JCAPI].

[1] Please note that, although the TOE supports other unlisted lengths (for legacy applications), some combinations algorithms/key sizes are considered not secure by SOG-IS (SOG-IS Crypto Evaluation Scheme Agreed Cryptographic Mechanisms, Version 1.3, February 2023) and should not be used to handle sensitive data.

6.1.1.2.3 FCS_COP.1 Cryptographic operation

FCS_COP.1.1 The TSF shall perform [assignment: *list of cryptographic operations in table below*] in accordance with a specified cryptographic algorithm [assignment: *cryptographic algorithm in table below*] and cryptographic key sizes [assignment: cryptographic key sizes in table below] that meet the following: [assignment: *list of standards in table below*].

Application Note:

3. The TOE shall provide a subset of cryptographic operations defined in [JCAPI] (see javacardx.crypto.Cipher and javacardx.security packages).
4. This component shall be instantiated according to the version of the Java Card API applicable to the security target and the implemented algorithms [JCAPI].

Iteration	Cryptographic operation	Cryptographic algorithm	Supported key size	Standards
AES	Signature, signature's verification, encryption and decryption	AES with Modes CBC, GCM, CCM, and CMAC	128, 192 and 256 bits	FIPS PUB 197 SP800-38A (CBC) SP800-38B (CMAC) SP800-38C (CCM) SP800-38D (GCM)
DES	Signature, signature's verification, encryption and decryption	Single-key DES, 2-key and 3-key TDES in CBC mode	112 or 168 bits [2]	NIST SP 800-67 NIST SP 800-38A
RSA	RSA public key operation; RSA private key operation with CRT; EMSA PSS and PKCS1 signature scheme coding; RSA Key Encapsulation Method (KEM)	Rivest, Shamir & Adleman's	Up to 4096 bits Up to 4096 bits Up to 4096 bits Up to 4096 bits	PKCS #1 v2.1
ECC	Diffie-Hellman (ECDH) key agreement computation Digital signature algorithm (ECDSA) generation and verification)	Elliptic Curves Cryptography on GF(p) on curves in Weierstrass form	160, 192, 224, 256, 320, 384, 512, 521 bits	FIPS 186-4 ANSI X.9.62 section 7 NIST 800-56A
HASH	Hash	SHA-1 SHA-224	NA	FIPS 180-4

[2] Please note that, although the TOE supports other unlisted lengths (for legacy applications), some combinations algorithms/key sizes are considered not secure by SOG-IS (SOG-IS Crypto Evaluation Scheme Agreed Cryptographic Mechanisms, Version 1.3, February 2023) and should not be used to handle sensitive data.

Iteration	Cryptographic operation	Cryptographic algorithm	Supported key size	Standards
		SHA-256 SHA-384 SHA-512 Protected SHA-1 Protected SHA-256 Protected SHA-384 Protected SHA-512		
HMAC	Signature	SHA-1 SHA-224 SHA-256 SHA-384 SHA-512 Protected SHA-1 Protected SHA-256 Protected SHA-384 Protected SHA-512	NA	FIPS 198-1
DH	Key agreement	Diffie-Hellman	(2048, 224) (2048, 256)	ANSI X9.42
CTR-RBG	CTR-RBG	AES	128, 192 and 256 bits	NIST SP 800-90A FIPS 197
Secure Messaging	Encryption and MAC computation	DES, AES	DES, AES key lengths	[ICAO] Doc 9303, Part 11

Application Note: some cryptographic algorithms can be disabled during personalization.

6.1.1.2.4 FCS_RNG.1/IC Random number generation

FCS_RNG.1.1/IC The TSF shall provide a [selection: *physical*] random number generator [selection: *PTG.2*] that implements: [assignment:

1. *(PTG.2.1) A total failure test detects a total failure of entropy source immediately when the RNG has started. When a total failure is detected, no random numbers will be output.*
2. *(PTG.2.2) If a total failure of the entropy source occurs while the RNG is being operated, the RNG prevents the output of any internal random number that depends on some raw random numbers that have been generated after the total failure of the entropy source.*
3. *(PTG.2.3) The online test shall detect non-tolerable statistical defects of the raw random number sequence (i) immediately when the RNG has started, and (ii) while the RNG is being operated. The TSF must not output any random numbers before the power-up online test has finished successfully or when a defect has been detected.*
4. *(PTG.2.4) The online test procedure shall be effective to detect non-tolerable weaknesses of the random numbers soon.*

5. *(PTG.2.5) The online test procedure checks the quality of the raw random number sequence. It is triggered externally. The online test is suitable for detecting non-tolerable statistical defects of the statistical properties of the raw random numbers within an acceptable period of time.*

].

FCS_RNG.1.2/IC The TSF shall provide random numbers that meet [assignment:

6. *(PTG.2.6) Test procedure A does not distinguish the internal random numbers from output sequences of an ideal RNG.*
7. *(PTG.2.7) The average Shannon entropy per internal random bit exceeds 0.997.*

].

Application Note:

5. The keys are reset as specified in [JCAPI] Key class, with the method clearKey(). Any access to a cleared key for ciphering or signing shall throw an exception.
6. This component shall be instantiated according to the version of the Java Card API applicable to the security target and the implemented algorithms [JCAPI].

6.1.1.2.5 FCS_RNG.1/DRBG Deterministic Random number generation

FCS_RNG.1.1/DRBG: The TSF shall provide a [selection: *deterministic*] random number generator [selection: *DRG.3*] that implements[assignment: *a list of security capabilities fulfilling requirements for Class DRG.3 defined in [BSI_AIS20/AIS31] standard:*
(DRG.3.1) if initialized with a random seed using a PTRNG of class PTG.2 as random source, the internal state of the RNG shall have at least 100 bits of min-entropy.
(DRG.3.2) The RNG provides forward secrecy
(DRG.3.3) The RNG provides backward secrecy even if the current internal state is known.
]

FCS_RNG.1.2/DRBG: The TSF shall provide random numbers that meet [assignment:
(DRG.3.4) The RNG initialized with a random seed during every startup and after 2^{32} requests, generates output for more than 2^{34} strings of bit length 128 that are mutually different with probability of $w>1-2^{-16}$.
(DRG.3.5) Statistical test suites cannot practically distinguish the random numbers from output sequences of an ideal RNG. The random numbers must pass test procedure A and the FIPS 140-2 statistical test suite.
]

6.1.1.3 Card Security Management

6.1.1.3.1 FAU_ARP.1 Security alarms

FAU_ARP.1.1 The TSF shall take **one of the following actions**:

1. throw an exception,

2. lock the card session,
3. reinitialize the Java Card System and its data,
4. [assignment: *none*]
upon detection of a potential security violation.

Refinement:
The "potential security violation" stands for one of the following events:
1. CAP file inconsistency,
2. typing error in the operands of a bytecode,
3. applet life cycle inconsistency,
4. card tearing (unexpected removal of the Card out of the CAD) and power failure, abort of a transaction in an unexpected context, (see abortTransaction(), [JCAPI] and ([JCRE], §7.6.2)
5. violation of the Firewall or JCVM SFPs,
6. unavailability of resources,
7. array overflow,
8. [assignment: *integrity error caused by a perturbation attack*].

Application Note:
1. The developer shall provide the exhaustive list of actual potential security violations the TOE reacts to. For instance, other runtime errors related to applet's failure like uncaught exceptions.
2. The bytecode verification defines a large set of rules used to detect a "potential security violation". The actual monitoring of these "events" within the TOE only makes sense when the bytecode verification is performed on-card.
3. Depending on the context of use and the required security level, there are cases where the card manager and the TOE must work in cooperation to detect and appropriately react in case of potential security violation. This behavior must be described in this component. It shall detail the nature of the feedback information provided to the card manager (like the identity of the offending application) and the conditions under which the feedback will occur (any occurrence of the java.lang.SecurityException exception).
4. The "locking of the card session" may not appear in the policy of the card manager. Such measure should only be taken in case of severe violation detection; the same holds for the re-initialization of the Java Card System. Moreover, the locking should occur when "clean" re-initialization seems to be impossible.
5. The locking may be implemented at the level of the Java Card System as a denial of service (through some systematic "fatal error" message or return value) that lasts up to the next "RESET" event, without affecting other components of the card (such as the card manager). Finally, because the installation of applets is a sensitive process, security alerts in this case should also be carefully considered herein.

6.1.1.3.2 FDP_SDI.2/DATA Stored data integrity monitoring and action

FDP_SDI.2.1/DATA The TSF shall monitor user data stored in containers controlled by the TSF for [assignment: *integrity errors*] on all objects, based

on the following attributes: [assignment: *integrity protected data*].

FDP_SDI.2.2/DATA Upon detection of a data integrity error, the TSF shall [assignment: *write a security error information persistently and mute the card*].

Application Note:

1. *Although no such requirement is mandatory in the Java Card specification, at least an exception shall be raised upon integrity errors detection on cryptographic keys, PIN values and their associated security attributes. Even if all the objects cannot be monitored, cryptographic keys and PIN objects shall be considered with particular attention by ST authors as they play a key role in the overall security.*
2. *It is also recommended to monitor integrity errors in the code of the native applications and Java Card applets.*
3. *For integrity sensitive application, their data shall be monitored (D.APP_I_DATA): applications may need to protect information against unexpected modifications, and explicitly control whether a piece of information has been changed between two accesses. For example, maintaining the integrity of an electronic purse's balance is extremely important because this value represents real money. Its modification must be controlled, for illegal ones would denote an important failure of the payment system.*
4. *A dedicated library could be implemented and made available to developers to achieve better security for specific objects, following the same pattern that already exists in cryptographic APIs, for instance.*

6.1.1.3.3 FPR_UNO.1 Unobservability

FPR_UNO.1.1 The TSF shall ensure that [assignment: *all users*] are unable to observe the operation [assignment: *all operations*] on [assignment: D.*APP_KEYs, D.PIN*] by [assignment: *all other users*]

Application Note:

The non-observability of operations on sensitive information such as keys appears as impossible to circumvent in the smart card world. The precise list of operations and objects is left unspecified, but should at least concern secret keys and PIN values when they exist on the card, as well as the cryptographic operations and comparisons performed on them

.

6.1.1.3.4 FPT_TDC.1 Inter-TSF basic TSF data consistency

FPT_TDC.1.1 The TSF shall provide the capability to consistently interpret **the CAP files, the bytecode and its data arguments** when shared between the TSF and another trusted IT product.

FPT_TDC.1.2 The TSF shall use

9. **the rules defined in [JCVM] specification,**
10. **the API tokens defined in the export files of reference implementation,**

1. [assignment: *none*]

when interpreting the TSF data from another trusted IT product.

Application Note:

Concerning the interpretation of data between the TOE and the underlying Java Card platform, it is assumed that the TOE is developed consistently with the SCP functions, including memory management, I/O functions and cryptographic functions.

6.1.1.4 AID Management

6.1.1.4.1 FIA_USB.1/AID User-subject binding

11. FIA_USB.1.1/AID The TSF shall associate the following user security attributes with subjects acting on the behalf of that user: **CAP file AID**.
12. FIA_USB.1.2/AID The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users: [assignment: *each CAP file is associated with a unique CAP file AID*].
13. FIA_USB.1.3/AID The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: [assignment: *the initially assigned CAP file AID in unchangeable*].

Application Note: The user is the applet and the subject is the S.CAP_FILE. The subject security attribute "Context" shall hold the user security attribute "CAP file AID".

6.1.2 InstG Security Functional Requirements

The following table shows all the SFRs from Java Card PP PP-JC that do not require to perform any operation and therefore are an exact copy of the PP. SFRs containing operations that have not been performed by the Java Card PP PP-JC are addressed in the following sections.

Section	SFR
InstG SFRs	FDP_ITC.2/Installer Import of user data with security attributes
	FMT_SMR.1/Installer Security roles
	FPT_FLS.1/Installer Failure with preservation of secure state

6.1.2.1 FPT_RCV.3/Installer Automated recovery without undue loss

FPT_RCV.3.1/Installer When automated recovery from [assignment: *none*] is not possible, the TSF shall enter a maintenance mode where the ability to return to a secure state is provided.

FPT_RCV.3.2/Installer For [assignment: *interrupted deletion, interrupted load or interrupted install (except if the register method has already been invoked)*], the TSF shall ensure the return of the TOE to a secure state using automated procedures.

FPT_RCV.3.3/Installer	The functions provided by the TSF to recover from failure or service discontinuity shall ensure that the secure initial state is restored without exceeding [assignment: *0%*] for loss of TSF data or objects under the control of the TSF.
FPT_RCV.3.4/Installer	The TSF shall provide the capability to determine the objects that were or were not capable of being recovered.

Application Note:

FPT_RCV.3.1/Installer:

1. This element is not within the scope of the Java Card specification, which only mandates the behavior of the Java Card System in good working order. Further details on the "maintenance mode" shall be provided in specific implementations. The following is an excerpt from [CC2], p298: In this maintenance mode normal operation might be impossible or severely restricted, as otherwise insecure situations might occur. Typically, only authorised users should be allowed access to this mode but the real details of who can access this mode is a function of FMT: Security management. If FMT: Security management does not put any controls on who can access this mode, then it may be acceptable to allow any user to restore the system if the TOE enters such a state. However, in practice, this is probably not desirable as the user restoring the system has an opportunity to configure the TOE in such a way as to violate the SFRs.

FPT_RCV.3.2/Installer:

2. Should the installer fail during loading/installation of a package/applet, it has to revert to a "consistent and secure state". The Java Card RE has some clean up duties as well; see [JCRE], §11.1.5 for possible scenarios. Precise behavior is left to implementers. This component shall include among the listed failures the deletion of a package/applet. See ([JCRE], 11.3.4) for possible scenarios. Precise behavior is left to implementers.
3. Other events such as the unexpected tearing of the card, power loss, and so on, are partially handled by the underlying hardware platform (see [PP0035]) and, from the TOE's side, by events "that clear transient objects" and transactional features. See FPT_FLS.1.1, FDP_RIP.1/TRANSIENT, FDP_RIP.1/ABORT and FDP_ROL.1/FIREWALL.

FPT_RCV.3.3/Installer:

The quantification is implementation dependent, but some facts can be recalled here. First, the SCP ensures the atomicity of updates for fields and objects, and a power-failure during a transaction or the normal runtime does not create the loss of otherwise permanent data, in the sense that memory on a smart card is essentially persistent with this respect (EEPROM). Data stored on the RAM and subject to such failure is intended to have a limited lifetime anyway (runtime data on the stack, transient objects' contents). According to this, the loss of data within the TSF scope should be limited to the same restrictions of the transaction mechanism.

6.1.3 ADELG Security Functional Requirements

The following table shows all the SFRs from Java Card PP PP-JC that do not require to perform any operation and therefore are an exact copy of the PP. SFRs containing operations that have not been performed by the Java Card PP PP-JC are addressed in the following sections.

Section	SFR
ADELG SFRs	FDP_ACC.2/ADEL Complete access control
	FDP_ACF.1/ADEL Security attribute based access control
	FDP_RIP.1/ADEL Subset residual information protection
	FMT_MSA.1/ADEL Management of security attributes
	FMT_MSA.3/ADEL Static attribute initialisation
	FMT_SMF.1/ADEL Specification of Management Functions
	FMT_SMR.1/ADEL Security roles
	FPT_FLS.1/ADEL Failure with preservation of secure state

6.1.4 ODELG Security Functional Requirements

The following table shows all the SFRs from Java Card PP PP-JC that do not require to perform any operation and therefore are an exact copy of the PP. This section does not contain any SFRs with operations still to be performed.

Section	SFR
ODELG SFRs	FDP_RIP.1/ODEL Subset residual information protection
	FPT_FLS.1/ODEL Failure with preservation of secure state

6.1.5 CarG Security Functional Requirements

The following table shows all the SFRs from Java Card PP PP-JC that do not require to perform any operation and therefore are an exact copy of the PP. SFRs containing operations that have not been performed by the Java Card PP PP-JC are addressed in the following sections.

Section	SFR
CarG SFRs	FDP_IFC.2/CM Complete information flow control
	FTP_ITC.1/CM Inter-TSF trusted channel

6.1.5.1 FCO_NRO.2/CM Enforced proof of origin

FCO_NRO.2.1/CM	The TSF shall enforce the generation of evidence of origin for transmitted **application CAP files** at all times.
FCO_NRO.2.2/CM [Editorially Refined]	The TSF shall be able to relate the **identity** of the originator of the information, and the **application CAP file,** of the information to which the evidence applies.

FCO_NRO.2.3/CM The TSF shall provide a capability to verify the evidence of origin of information to **recipient** given [assignment: *at the time the Executable load files are received as no evidence is kept on the card for future verification*].

Application Note:

FCO_NRO.2.1/CM:

1. Upon reception of a new application CAP file for installation, the card manager shall first check that it actually comes from the verification authority and represented by the subject S.BCV. The verification authority is indeed the entity responsible for bytecode verification.

FCO_NRO.2.3/CM:

2. The exact limitations on the evidence of origin are implementation dependent. In most of the implementations, the card manager performs an immediate verification of the origin of the CAP file using an electronic signature mechanism, and no evidence is kept on the card for future verifications.

6.1.5.2 FDP_IFF.1/CM Simple security attributes

FDP_IFF.1.1/CM The TSF shall enforce the **CAP FILE LOADING information flow control SFP** based on the following types of subject and information security attributes: [assignment: *Load file, DAP authenticated, OTA authenticated*].

FDP_IFF.1.2/CM The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold: [assignment: *the rules describing the communication protocol used by the CAD and the card for transmitting a new package as detailed in [GP] Section 9.3.9*].

FDP_IFF.1.3/CM The TSF shall enforce the [assignment: *none*].

FDP_IFF.1.4/CM The TSF shall explicitly authorise an information flow based on the following rules: [assignment: *none*].

FDP_IFF.1.5/CM The TSF shall explicitly deny an information flow based on the following rules:

1. **The TOE fails to verify the integrity and authenticity evidences of the application CAP file.**
2. [assignment: *the rules describing the communication protocol used by the CAD and the card for transmitting a new package as detailed in [GP] Section 9.3.9*].

Application Note:
FDP_IFF.1.1/CM:

1. The security attributes used to enforce the CAP FILE LOADING SFP are implementation dependent. More precisely, they depend on the communication protocol enforced between the CAD and the card. For instance, some of the attributes that can be used are: (1) the keys used by the subjects to encrypt/decrypt their messages; (2) the number of pieces the application package has been split into in order to be sent to the card; (3) the ordinal of each piece in the decomposition of the package, etc. See for example Appendix D of [GP].

FDP_IFF.1.2/CM:

2. The precise set of rules to be enforced by the function is implementation dependent. The whole exchange of messages shall verify at least the following two rules: (1) the subject S.INSTALLER shall accept a message only if it comes from the subject S.CAD; (2) the subject S.INSTALLER shall accept an application package only if it has received without modification and in the right order all the APDUs sent by the subject S.CAD.

FDP_IFF.1.5/CM:

3. The verification of the integrity and authenticity evidences can be performed either during loading or during the first installation of an application of the CAP file.

6.1.5.3 FDP_UIT.1/CM Data exchange integrity

FDP_UIT.1.1/CM	The TSF shall enforce the **CAP FILE LOADING information flow control SFP** to [selection: *receive*] user data in a manner protected from [selection: *modification, deletion, insertion, replay*] errors.
FDP_UIT.1.2/CM [Editorially Refined]	The TSF shall be able to determine on receipt of user data, whether **modification, deletion, insertion, replay of some of the pieces of the application sent by the CAD** has occurred.

Application Note:

Modification errors should be understood as modification, substitution, unrecoverable ordering change of data and any other integrity error that may cause the application CAP file to be installed on the card to be different from the one sent by the CAD.

6.1.5.4 FIA_UID.1/CM Timing of identification

FIA_UID.1.1/CM	The TSF shall allow [assignment: 4. *application selection* *5. initializing a secure channel with the card* *6. requesting data that identifies the card or the Card Issuer*] on behalf of the user to be performed before the user is identified.
FIA_UID.1.2/CM	The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

Application Note:
The list of TSF-mediated actions is implementation-dependent, but CAP file installation requires the user to be identified. Here by user is meant the one(s) that in the Security Target shall be associated to the role(s) defined in the component FMT_SMR.1/CM.

6.1.5.5 FMT_MSA.1/CM Management of security attributes

FMT_MSA.1.1/CM	The TSF shall enforce the **CAP FILE LOADING information flow control SFP** to restrict the ability to [selection: *modify* [assignment: *no other operations*]] the security attributes [assignment: *key data, card life cycle state, secure configuration, default SELECTED configuration*] to [assignment: *card manager*].

6.1.5.6 FMT_MSA.3/CM Static attribute initialisation

FMT_MSA.3.1/CM	The TSF shall enforce the **CAP FILE LOADING information flow control SFP** to provide **restrictive** default values for security attributes that are used to enforce the SFP.
FMT_MSA.3.2/CM	The TSF shall allow the [assignment: *card manager*] to specify alternative initial values to override the default values when an object or information is created.

6.1.5.7 FMT_SMF.1/CM Specification of Management Functions

FMT_SMF.1.1/CM	The TSF shall be capable of performing the following management functions: [assignment: *key data, card life cycle state, secure configuration, default SELECTED configuration*].

6.1.5.8 FMT_SMR.1/CM Security roles

FMT_SMR.1.1/CM	The TSF shall maintain the roles [assignment: *card manager*].
FMT_SMR.1.2/CM	The TSF shall be able to associate users with roles.

6.1.6 Additional Security Functional Requirements

6.1.6.1 FPT_TST.1 TSF Testing

FPT_TST.1.1	The TSF shall run a suite of self tests [selection: *[assignment: at the conditions: during start-up and periodically during normal operation]*] to demonstrate the correct operation of the [selection: *TSF*].
FPT_TST.1.2	The TSF shall provide authorized users with the capability to verify the integrity of [selection: *TSF Data*].

FPT_TST.1.3	The TSF shall provide authorized users with the capability to verify the integrity of [selection: *[assignment: parts of TSF (TSF executable code)]*].

6.1.7 Optional package: Sensitive Results

6.1.7.1 FDP_SDI.2/RESULT Integrity_Sensitive_Result

FDP_SDI.2.1/RESULT	The TSF shall monitor user data stored in containers controlled by the TSF for **integrity errors** on all objects, based on the following attributes: **sensitive API result stored in the javacardx.security.SensitiveResult class**.
FDP_SDI.2.2/RESULT	Upon detection of a data integrity error, the TSF shall **throw an exception**.

Application Note:
This requirement applies in particular to the results stored by the javacardx.security.SensitiveResult class (if supported).

6.2 **Firmware Upgrade OS**

Application note: Firmware upgrade can be disabled during personalization. In case it is disabled the following SFRs do not apply.

6.2.1 FTP_ITC.1/Loader Inter-TSF trusted channel

FTP_ITC.1.1/Loader	The TSF shall provide a communication channel between itself and ***the Loader user*** that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure.
FTP_ITC.1.2/Loader	The TSF shall permit [selection: *another trusted IT product*] to initiate communication via the trusted channel.
FTP_ITC.1.3/Loader	The TSF shall initiate communication via the trusted channel for [assignment: *enabling Secure Loader to perform loading operation*].

6.2.2 FDP_ACC.1/Loader Subset access control – Loader

FDP_ACC.1.1/Loader	The TSF shall enforce the [assignment: *Loader SFP*] on [assignment: 1. *the subjects: Loader user,* 2. *the objects: software image data,* 3. *the operation: performing a software loading*]

6.2.3 FDP_ACF.1/Loader Security attribute based access control – Loader

FDP_ACF.1.1/Loader	The TSF shall enforce the [assignment: *Loader SFP*] to objects based on the following [assignment: 1. *the subjects: Loader user with security attributes "Authenticated",* 2. *the objects: software image data in memory with security attributes none.*]
FDP_ACF.1.2/Loader	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed [assignment: 1. *the Loader user with security attribute "Authenticated" set to "yes" can access to Secure Loader to perform a software loading operation.*]

FDP_ACF.1.3/Loader	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules [assignment: *none*].
FDP_ACF.1.4/Loader	The TSF shall explicitly deny access of subjects to objects based on the following additional rules [assignment: *1. the Loader user with security attribute "Authenticated" set to "no" cannot access to Secure Loader to perform a software loading operation.*]

7 Security Assurance Requirement

This Security Target claims conformance to EAL6 augmented with ALC_FLR.2.

ADV_ARC is refined.

ADV_SPM.1 Formal TOE security policy model.

ADV_SPM.1.1D: The Developer shall provide a formal security policy model for the **[assignment: FIREWALL access control SFP and JCVM information flow control SFP]**.

The requirements are summarised in the following table:

Assurance Class	Component	Component Title
ADV Development	ADV_ARC.1	Security architecture NOTE: This component has been refined as follows: *ADV_ARC.1.2C The security architecture description shall describe the security domains maintained by the TSF consistently with the SFRs.* *Refinement:* *In particular, the TOE shall maintain the applet isolation without requiring more rules on applet verification than the* [GP-SGBA].
	ADV_FSP.5	Complete semi-formal functional specification with additional error information
	ADV_IMP.2	Implementation representation of the TSF
	ADV_INT.3	Minimally complex internals
	ADV_TDS.5	Complete Semiformal modular design
	ADV_SPM.1	Formal TOE Security Policy Model
AGD: Guidance documents	AGD_OPE.1	Operational user guidance
	AGD_PRE.1	Preparative procedures
ALC_ Life-cycle support	ALC_CMC.5	Advanced support
	ALC_CMS.5	Development tools CM coverage
	ALC_DEL.1	Delivery procedures
	ALC_DVS.2	Sufficiency of security measures
	ALC_LCD.1	Developer defined life-cycle model
	ALC_TAT.3	Compliance with implementation standards - all parts
	ALC_FLR.2	Flaw reporting procedures
ASE: Security Target evaluation	ASE_CCL.1	Conformance claims
	ASE_ECD.1	Extended components definition
	ASE_INT.1	ST introduction
	ASE_TSS.1	TOE summary specification
	ASE_OBJ.2	Security objectives
	ASE_REQ.2	Derived security requirements
	ASE_SPD.1	Security problem definition
	ASE_TSS.1	TOE summary specification
ATE: Tests	ATE_COV.3	Rigorous analysis of coverage
	ATE_DPT.3	Testing: modular design

Assurance Class	Component	Component Title
	ATE_FUN.2	Ordered functional testing
	ATE_IND.2	Independent testing
AVA: Vulnerability assessment	AVA_VAN.5	Advanced methodical vulnerability analysis

Table 16 EAL6 requirements description extended with augmented with ALC_FLR.2

8 TOE Summary Specification

8.1 Security Functionality

8.1.1 Java Card

SF.FIREWALL	The TOE implements an applet firewall according to [JCRE]. Each applet on the TOE must have been passed the Bytecode Verifier in order to ensure correct applet isolation. As an additional defensive security feature also a type check for API array parameters is performed. This TSF enforces the following SFRs: 1. FDP_ACC.2/FIREWALL Complete access control 2. FDP_ACF.1/FIREWALL Security attribute based access control 3. FDP_IFC.1/JCVM Subset information flow control 4. FDP_IFF.1/JCVM Simple security attributes 5. FMT_MSA.1/JCRE Management of security attributes 6. FMT_MSA.2/FIREWALL_JCVM Secure security attributes 7. FMT_MSA.3/FIREWALL Static attribute initialisation 8. FMT_MSA.3/JCVM Static attribute initialization 9. FMT_SMR.1 Security roles 10. FDP_ROL.1/FIREWALL Basic rollback 11. FMT_MSA.1/JCVM Management of security attributes 12. FMT_MTD.1/JCRE Management of TSF data 13. FMT_MTD.3/JCRE Secure TSF data 14. FMT_SMF.1 Specification of Management Functions
SF.RIP	The TOE ensures that sensitive information is made unavailable after deletion. This will be done by overwriting keys, APDU buffer and transient objects with zeros or random values. Applications and persistent objects will be marked as deleted. If the deleted resource is reused by a new object creation, the previous content will be set to a random value. This TSF enforces the following SFRs: 15. FDP_RIP.1/bArray Subset residual information protection 16. FDP_RIP.1/APDU Subset residual information protection 17. FDP_RIP.1/KEYS Subset residual information protection 18. FDP_RIP.1/TRANSIENT Subset residual information protection 19. FDP_RIP.1/ADEL Subset residual information protection 20. FDP_RIP.1/ODEL Subset residual information protection 21. FDP_RIP.1/ABORT Subset residual information protection 22. FDP_RIP.1/OBJECTS Subset residual information protection 23. FDP_RIP.1/GlobalArray Subset residual information protection
SF.Rollback	The TOE implements atomicity and rollback mechanism for Java Card runtime environment [JCRE] and GlobalPlatform management functions (see [GP]). The TOE also ensures that objects created during an aborted transaction are made unavailable. This TSF enforces the following SFRs: 24. FPT_RCV.3/Installer Automated recovery without undue loss 25. FDP_ROL.1/FIREWALL Basic rollback 26. FDP_RIP.1/ABORT Subset residual information protection
SF.SCP	The TOE implements secure channel protocols according to [GP], chapter 10. The following protocols are supported: 27. SCP03 according to [GP-D]. 28. Executable Load File Upgrade [GP-H]. The SCP uses as the basic cryptographic primitives the security hardened symmetric cryptographic library. This TSF enforces the following SFRs: 29. FDP_UIT.1/CM Data exchange integrity 30. FTP_ITC.1/CM Inter-TSF trusted channel 31. FCO_NRO.2/CM Enforced proof of origin 32. FDP_IFC.2/CM Complete information flow control

	33. FDP_IFF.1/CM Simple security attributes 34. FMT_MSA.1/CM Management of security attributes 35. FMT_MSA.3/CM Static attribute initialisation 36. FMT_SMF.1/CM Specification of Management Functions 37. FIA_UID.1/CM Timing of identification 38. FMT_SMR.1/CM Security roles 39. FCS_COP.1 Cryptographic operation
SF.CM	The TOE implements an access control policy for GlobalPlatform card management functions according to [GP] , D [GP-D]. In addition to the GP specification, the Java Card Runtime Environment specification [JCRE] is followed to support for application loading, installation, and deletion. AID management is provided by SF.CM according to the GlobalPlatform Specification [GP], the Java Card Runtime Environment Specification [JCRE], and the Java Card API Specification [JCAPI]. This TSF enforces the following SFRs: 40. FMT_MSA.1/CM Management of security attributes 41. FMT_MSA.3/CM Static attribute initialisation 42. FMT_SMF.1/CM Specification of Management Functions 43. FMT_SMR.1/CM Security roles 44. FPT_TDC.1 Inter-TSF basic TSF data consistency 45. FIA_ATD.1/AID User attribute definition 46. FIA_UID.2/AID User identification before any action 47. FIA_USB.1/AID User-subject binding 48. FDP_ITC.2/Installer Import of user data with security attributes 49. FMT_SMR.1/Installer Security roles 50. FPT_RCV.3/Installer Automated recovery without undue loss 51. FPT_FLS.1/Installer Failure with preservation of secure state 52. FDP_ACC.2/ADEL Complete access control 53. FDP_ACF.1/ADEL Security attribute based access control 54. FDP_RIP.1/ADEL Subset residual information protection 55. FMT_MSA.1/ADEL Management of security attributes 56. FMT_MSA.3/ADEL Static attribute initialisation 57. FMT_SMR.1/ADEL Security roles 58. FPT_FLS.1/ADEL Failure with preservation of secure state 59. FMT_SMF.1/ADEL Specification of Management Functions 60. FPT_FLS.1/ODEL Failure with preservation of secure state
SF.Physical	The TOE provides means to protect SFRs against physical tampering and leakage. The TOE uses mainly the physical security measures of the underlying hardware platform. Security mechanisms involved in this protection are: 1. Memories scrambling and encryption 2. Protection of NVM sectors 3. Memory Protection Unit (MPU) 4. Library Protection Unit (LPU) This TSF enforces the following SFRs: 61. FAU_ARP.1 Security alarms 62. FDP_SDI.2/DATA Stored data integrity monitoring and action 63. FPT_TST.1 TSF testing 64. FPT_FLS.1 Failure with preservation of secure state
SF.CRYPTO	The TOE provides key creation, key management, key deletion and cryptographic functionality. It provides the API in accordance to the Java Card API Specification [JCAPI]. The cryptographic API uses as the basic cryptographic implementation the security hardened cryptographic library which is CC certified together with the underlying platform. The integrity of the cryptographic assets is monitored. In addition, key destructions and residual information purging is implemented. SF.CRYPTO provides secure random number generation and makes this functionality available through an API according to the Java Card API Specification [JCAPI]. In addition SF.CRYPTO provides encryption and MAC computation using DES and AES algorithms as outlined in [ICAO] 9303, Part 11 for Machine Readable Travel Documents through proprietary API for Secure Messaging. This TSF enforces the following SFRs:

	65. FCS_CKM.1 Cryptographic key generation 66. FCS_CKM.4 Cryptographic key destruction 67. FCS_COP.1 Cryptographic operation 68. FPR_UNO.1 Unobservability 69. FDP_SDI.2/DATA Stored data integrity monitoring and action 70. FCS_RNG.1/IC Physical Random number generation 71. FCS_RNG.1/DRBG Deterministic Random number generation
SF.PIN	The TOE implements secure PIN compare functions and integrity protection of the PIN. This TSF enforces the following SFRs: 72. FPR_UNO.1 Unobservability 73. FDP_SDI.2/DATA Stored data integrity monitoring and action
SF.SENSITIVE_RESULTS	The TOE implements the sensitive results and makes this functionality available through the API in `javacardx.security.SensitiveResults` according to the Java Card API Specification [JCAPI]. This TSF enforces the following SFR: 74. FDP_SDI.2/RESULT

8.1.2 Firmware Upgrade Functionality

SF.AUTH-LOADER	For Loader User, authentication is achieved through the establishment of the secure channel with the Firmware Upgrade Enablement applet, which relies on the GP SCP protocols supported by its associated Security domain. The secure channel can be established using a dedicated Loader user key set. The TOE enforces access control by verifying that specific actions are authorized by the proper credential. Establishment of the secure channel with the proper key sets provides to the Loader user the rights to enable the Secure Loader functionality. Firmware Upgrade OS security operations are authorized to Loader User. This functionality meets the SFR related to user authentication and access control: Firmware Upgrade OS functionalities authentication and access control: 75. FTP_ITC.1/Loader 76. FDP_ACC.1/Loader 77. FDP_ACF.1/Loader

9 Rationales

9.1 **Conformance Claim Rationale**

The statement of security functional requirements copies all SFRs as defined in the PP [PP-JC].

9.2 **Security Requirements Rationale**

9.2.1 Java Card

Objective	Rationale
O.SID	Subjects' identity is AID-based (applets, packages and CAP files), and is met by the following SFRs: FDP_ITC.2/Installer, FIA_ATD.1/AID, FMT_MSA.1/JCRE, FMT_MSA.1/JCVM, FMT_MSA.1/ ADEL, FMT_MSA.1/CM, FMT_MSA.3/ADEL, FMT_MSA.3/FIREWALL, FMT_MSA.3/JCVM, FMT_MSA.3/CM, FMT_SMF.1/CM, FMT_SMF.1/ADEL, FMT_MTD.1/JCRE and FMT_MTD.3/JCRE. Installation procedures ensure protection against forgery (the AID of an applet is under the control of the TSFs) or re-use of identities (FIA_UID.2/AID, FIA_USB.1/AID).
O.FIREWALL	This objective is met by the FIREWALL access control policy FDP_ACC.2/FIREWALL and FDP_ACF.1/FIREWALL, the JCVM information flow control policy (FDP_IFF.1/JCVM, FDP_IFC.1/JCVM) and the functional requirement FDP_ITC.2/Installer. The functional requirements of the class FMT (FMT_MTD.1/JCRE, FMT_MTD.3/JCRE, FMT_SMR.1/Installer, FMT_SMR.1, FMT_SMF.1, FMT_SMR.1/ADEL, FMT_SMF.1/ADEL, FMT_SMF.1/CM, FMT_MSA.1/CM, FMT_MSA.3/CM, FMT_SMR.1/CM, FMT_MSA.2/FIREWALL_JCVM, FMT_MSA.3/FIREWALL, FMT_MSA.3/JCVM, FMT_MSA.1/ADEL, FMT_MSA.3/ADEL, FMT_MSA.1/JCRE, FMT_MSA.1/JCVM) also indirectly contribute to meet this objective.
O.GLOBAL_ARRAYS_CONFID	Only arrays can be designated as global, and the only global arrays required in the Java Card API are the APDU buffer, the global byte array input parameter (bArray) to an applet's install method and the global arrays created by the JCSystem.makeGlobalArray(…) method. The clearing requirement of these arrays is met by (FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray and FDP_RIP.1/bArray respectively). The JCVM information flow control policy (FDP_IFF.1/JCVM, FDP_IFC.1/JCVM) prevents an application from keeping a pointer to a shared buffer, which could be used to read its contents when the buffer is being used by another application.
O.GLOBAL_ARRAYS_INTEG	This objective is met by the JCVM information flow control policy (FDP_IFF.1/JCVM, FDP_IFC.1/JCVM), which prevents an application from keeping a pointer to the APDU buffer of the card, to the global byte array of the applet's install method or to the global arrays created by the JCSystem.makeGlobalArray(…) method. Such a pointer could be used to access and modify it when the buffer is being used by another application.
O.NATIVE	This security objective is covered by FDP_ACF.1/FIREWALL: the only means to execute native code is the invocation of a Java Card API method. This objective mainly relies on the environmental objective OE.CAP_FILE, which uphold the assumption A.CAP_FILE.
O.OPERATE	The TOE is protected in various ways against applets' actions (FPT_TDC.1), the FIREWALL access control policy FDP_ACC.2/FIREWALL and FDP_ACF.1/FIREWALL, and is able to detect and block various failures or security violations during usual working (FPT_FLS.1/ADEL, FPT_FLS.1, FPT_FLS.1/ODEL, FPT_FLS.1/Installer, FAU_ARP.1). Its security-critical parts and procedures are also protected: safe recovery from failure is ensured (FPT_RCV.3/Installer), applets' installation may be cleanly aborted (FDP_ROL.1/FIREWALL), communication with external users and their internal subjects is well-controlled (FDP_ITC.2/Installer, FIA_ATD.1/AID, FIA_USB.1/AID) to prevent alteration of TSF data (also protected by components of the FPT class). Almost every objective and/or functional requirement indirectly contributes to this one too.

Objective	Rationale
	Application note: Startup of the TOE (TSF-testing) can be covered by FPT_TST.1. This SFR component is not mandatory in [JCRE], but appears in most of security requirements documents for masked applications. Testing could also occur randomly. Self-tests may become mandatory in order to comply with FIPS certification [FIPS 140-2].
O.REALLOCATION	This security objective is satisfied by the following SFRs: FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray, FDP_RIP.1/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/TRANSIENT, FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/ADEL, which imposes that the contents of the re-allocated block shall always be cleared before delivering the block.
O.RESOURCES	The TSFs detects stack/memory overflows during execution of applications (FAU_ARP.1, FPT_FLS.1/ADEL, FPT_FLS.1, FPT_FLS.1/ODEL, FPT_FLS.1/Installer). Failed installations are not to create memory leaks (FDP_ROL.1/FIREWALL, FPT_RCV.3/Installer) as well. Memory management is controlled by the TSF (FMT_MTD.1/JCRE, FMT_MTD.3/JCRE, FMT_SMR.1/Installer, FMT_SMR.1, FMT_SMF.1 FMT_SMR.1/ADEL, FMT_SMF.1/ADEL, FMT_SMF.1/CM and FMT_SMR.1/CM).
O.ALARM	This security objective is met by FPT_FLS.1/Installer, FPT_FLS.1, FPT_FLS.1/ADEL, FPT_FLS.1/ODEL which guarantee that a secure state is preserved by the TSF when failures occur, and FAU_ARP.1 which defines TSF reaction upon detection of a potential security violation.
O.CIPHER	This security objective is directly covered by FCS_CKM.1, FCS_CKM.4 and FCS_COP.1. The SFR FPR_UNO.1 contributes in covering this security objective and controls the observation of the cryptographic operations which may be used to disclose the keys.
O.RNG	This security objective is directly covered by FCS_RNG.1/IC and FCS_RNG.1/DRBG which ensure the cryptographic quality of random number generation.
O.KEY-MNGT	This relies on the same security functional requirements as O.CIPHER, plus FDP_RIP.1 and FDP_SDI.2/DATA as well. Precisely it is met by the following components: FCS_CKM.1, FCS_CKM.4, FCS_COP.1, FPR_UNO.1, FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray FDP_RIP.1/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL and FDP_RIP.1/TRANSIENT.
O.PIN-MNGT	This security objective is ensured by FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray FDP_RIP.1/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL, FDP_RIP.1/TRANSIENT, FPR_UNO.1, FDP_ROL.1/FIREWALL and FDP_SDI.2/DATA security functional requirements. The TSFs behind these are implemented by API classes. The firewall security functions FDP_ACC.2/FIREWALL and FDP_ACF.1/FIREWALL shall protect the access to private and internal data of the objects.
O.TRANSACTION	Directly met by FDP_ROL.1/FIREWALL, FDP_RIP.1/ABORT, FDP_RIP.1/ODEL, FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray FDP_RIP.1/bArray, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL, FDP_RIP.1/TRANSIENT and FDP_RIP.1/OBJECTS (more precisely, by the element FDP_RIP.1.1/ABORT).
O.OBJ-DELETION	This security objective specifies that deletion of objects is secure. The security objective is met by the security functional requirements FDP_RIP.1/ODEL and FPT_FLS.1/ODEL.
O.DELETION	This security objective specifies that applet and CAP file deletion must be secure. The non-introduction of security holes is ensured by the ADEL access control policy (FDP_ACC.2/ADEL, FDP_ACF.1/ADEL). The integrity and confidentiality of data that does not belong to the deleted applet or CAP file is a by-product of this policy as well. Non-accessibility of deleted data is met by FDP_RIP.1/ADEL and the TSFs are protected against possible failures of the deletion procedures (FPT_FLS.1/ADEL, FPT_RCV.3/Installer). The security functional requirements of the class FMT (FMT_MSA.1/ADEL, FMT_MSA.3/ADEL, FMT_SMR.1/ADEL) included in the group ADELG also contribute to meet this objective.
O.LOAD	This security objective specifies that the loading of a CAP file into the card must be secure. Evidence of the origin of the CAP file is enforced (FCO_NRO.2/CM) and the integrity of the corresponding data is under the control of the CAP FILE LOADING information flow policy (FDP_IFC.2/CM, FDP_IFF.1/CM) and FDP_UIT.1/CM. Appropriate identification (FIA_UID.1/CM) and transmission mechanisms are also enforced (FTP_ITC.1/CM).

Objective	Rationale
O.INSTALL	This security objective specifies that installation of applets must be secure. Security attributes of installed data are under the control of the FIREWALL access control policy (FDP_ITC.2/Installer), and the TSFs are protected against possible failures of the installer (FPT_FLS.1/Installer, FPT_RCV.3/Installer).
O.SENSITIVE_RESULTS_INTEG	This security objective specifies that the sensitive results (`javacardx.security.SensitiveResults`) of sensitive operations executed by applications through the Java Card API are integrity protected specifically against physical attacks, and is fulfilled by FDP_SDI.2/RESULT.
O.CARD-MANAGEMENT	This objective is fulfilled by the following set of SFR: FDP_ACC.2/ADEL and FDP_ACF.1/ADEL contribute to meet the objective by the ADEL access control policy which ensures the non-introduction of security holes. FDP_RIP.1/ADEL ensures the non-accessibility of deleted data. FMT_MSA.1/ADEL and FMT_MSA.3/ADEL enforce the ADEL access control SFP. FMT_SMR.1/ADEL maintains the role applet deletion manager. FPT_RCV.3/Installer protects the TSFs against possible failures of the deletion procedures. FPT_FLS.1/Installer protects the TSFs against possible failures of the installer. FPT_FLS.1/ADEL protects the TSFs against possible failures of the deletion procedures. FDP_UIT.1/CM enforces the Secure Channel Protocol information flow control policy and the Security Domain access control policy which controls the integrity of the corresponding data. FDP_IFF.1/CM ensures the access control policy for the loaded data (as packages). The FCO_NRO.2/CM ensures the origin of the load file. It verifies the identity of the origin of the load file before start the loading FDP_IFC.2/CM ensures that loading commands are issued in the Secure Channel session. FDP_ROL.1/Firewall ensures that the card management operations are cleaned aborted FDP_ITC.2/Installer enforces the Firewall access control policy and flow control policy when importing card management data. FPT_FLS.1/ODEL ensures the preservation of secure state when failures occur. FMT_MSA.1/CM ensures the management of the security attributes to the card manager, for the modification of the defined security attributes. FMT_MSA.3/CM ensures that the security attributes can only be changed by the card manager. FMT_SMF.1/CM allows only the card manger to modify the security attributes of the management functions. The security role is specified in the FMT_SMR.1/CM. FTP_ITC.1/CM ensures the trusted Channel Communications. FPR_UNO.1 ensures the un-observability of the CM key when imported.. FPT_TST.1 ensures the correct operation of the card management functions as it tests the integrity of the TSF functions during initial start-up.
O.SCP.RECOVERY	SFR FAU_ARP.1 contributes to meet the objective by ensuring the reinitialization of the Java Card System and its data after card tearing and power failure. SFR FPT_FLS.1 contributes to meet the objective by preserving a secure state after failure.
O.SCP.SUPPORT	All crypto SFRs supports this objective as they provide the functionality to the Java Card and Global Platform (FCS_CKM.1, FCS_CKM.4, FCS_COP.1) Also SFR FDP_ROL.1/FIREWALL contributes to the realization of the objective.
O.SCP.IC	This objective is met by the physical protection given by the underlying IC plus the contribution of the following SFRs: 1. FPR_UNO.1, that contributes to the coverage of the objective by ensuring leakage resistant implementations of the unobservable operations and 2. FAU_ARP.1, that contributes to the coverage of the objective by resetting the card session or terminating the card in case of physical tampering.

9.2.2 Firmware Upgrade OS

SFR / TOE objectives	OT.ACCESS_CONTROL
FTP_ITC.1/Loader	Only valid authenticated users can access the Secure Loader functionality.This objective is met by the Loader SFP (FDP_ACC.1/Loader and FDP_ACF.1/Loader) in conjunction with FTP_ITC.1/Loader that ensures that the Secure Loader enabling is subject to the prior establishment of a Secure communication channel.
FDP_ACC.1/Loader	
FDP_ACF.1/Loader	

9.3 **Dependency Rationale**

9.3.1 Java Card

Requirement	Dependency	Satisfied by
FDP_ACC.2/FIREWALL	(FDP_ACF.1)	FDP_ACF.1/FIREWALL
FDP_ACF.1/FIREWALL	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.2/FIREWALL, FMT_MSA.3/FIREWALL
FDP_IFC.1/JCVM	(FDP_IFF.1)	FDP_IFF.1/JCVM
FDP_IFF.1/JCVM	(FDP_IFC.1) and (FMT_MSA.3)	FDP_IFC.1/JCVM, FMT_MSA.3/JCVM
FDP_RIP.1/OBJECTS	No Dependencies	
FMT_MSA.1/JCRE	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.2/FIREWALL, FMT_SMR.1
FMT_MSA.1/JCVM	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.2/FIREWALL, FDP_IFC.1/JCVM, FMT_SMF.1, FMT_SMR.1
FMT_MSA.2/FIREWALL_JCVM	(FDP_ACC.1 or FDP_IFC.1) and (FMT_MSA.1) and (FMT_SMR.1)	FDP_ACC.2/FIREWALL, FDP_IFC.1/JCVM, FMT_MSA.1/JCRE, FMT_MSA.1/JCVM, FMT_SMR.1
FMT_MSA.3/FIREWALL	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/JCRE, FMT_MSA.1/JCVM, FMT_SMR.1
FMT_MSA.3/JCVM	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/JCVM, FMT_SMR.1
FMT_SMF.1	No Dependencies	
FMT_SMR.1	(FIA_UID.1)	FIA_UID.2/AID
FCS_CKM.1	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_COP.1, FCS_CKM.4
FCS_CKM.4	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2)	FCS_CKM.1
FCS_COP.1	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1, FCS_CKM.4
FCS_RNG.1/IC	No Dependencies	
FCS_RNG.1/DRBG	No Dependencies	
FDP_RIP.1/ABORT	No Dependencies	
FDP_RIP.1/APDU	No Dependencies	
FDP_RIP.1/bArray	No Dependencies	
FDP_RIP.1/GlobalArray	No Dependencies	
FDP_RIP.1/KEYS	No Dependencies	

FDP_RIP.1/TRANSIENT	No Dependencies	
FDP_ROL.1/FIREWALL	(FDP_ACC.1 or FDP_IFC.1)	FDP_ACC.2/FIREWALL, FDP_IFC.1/JCVM
FAU_ARP.1	(FAU_SAA.1)	
FDP_SDI.2/DATA	No Dependencies	
FPR_UNO.1	No Dependencies	
FPT_FLS.1	No Dependencies	
FPT_TDC.1	No Dependencies	
FIA_ATD.1/AID	No Dependencies	
FIA_UID.2/AID	No Dependencies	
FIA_USB.1/AID	(FIA_ATD.1)	FIA_ATD.1/AID
FMT_MTD.1/JCRE	(FMT_SMF.1) and (FMT_SMR.1)	FMT_SMF.1, FMT_SMR.1
FMT_MTD.3/JCRE	(FMT_MTD.1)	FMT_MTD.1/JCRE
FDP_ITC.2/Installer	(FDP_ACC.1 or FDP_IFC.1) and (FPT_TDC.1) and (FTP_ITC.1 or FTP_TRP.1)	FDP_IFC.2/CM, FTP_ITC.1/CM, FPT_TDC.1
FMT_SMR.1/Installer	(FIA_UID.1)	
FPT_FLS.1/Installer	No Dependencies	
FPT_RCV.3/Installer	(AGD_OPE.1)	AGD_OPE.1
FDP_ACC.2/ADEL	(FDP_ACF.1)	FDP_ACF.1/ADEL
FDP_ACF.1/ADEL	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.2/ADEL, FMT_MSA.3/ADEL
FDP_RIP.1/ADEL	No Dependencies	
FMT_MSA.1/ADEL	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.2/ADEL, FMT_SMF.1/ADEL, FMT_SMR.1/ADEL
FMT_MSA.3/ADEL	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/ADEL, FMT_SMR.1/ADEL
FMT_SMF.1/ADEL	No Dependencies	
FMT_SMR.1/ADEL	(FIA_UID.1)	
FPT_FLS.1/ADEL	No Dependencies	
FDP_RIP.1/ODEL	No Dependencies	
FPT_FLS.1/ODEL	No Dependencies	
FCO_NRO.2/CM	(FIA_UID.1)	FIA_UID.1/CM
FDP_IFC.2/CM	(FDP_IFF.1)	FDP_IFF.1/CM
FDP_IFF.1/CM	(FDP_IFC.1) and (FMT_MSA.3)	FDP_IFC.2/CM, FMT_MSA.3/CM
FDP_UIT.1/CM	(FDP_ACC.1 or FDP_IFC.1) and (FTP_ITC.1 or FTP_TRP.1)	FDP_IFC.2/CM, FTP_ITC.1/CM
FIA_UID.1/CM	No Dependencies	
FMT_MSA.1/CM	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_IFC.2/CM, FMT_SMF.1/CM, FMT_SMR.1/CM
FMT_MSA.3/CM	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/CM, FMT_SMR.1/CM
FMT_SMF.1/CM	No Dependencies	
FMT_SMR.1/CM	(FIA_UID.1)	FIA_UID.1/CM
FTP_ITC.1/CM	No Dependencies	
FDP_SDI.2/RESULT	No Dependencies	
FPT_TST.1	No Dependencies	

Rationale for the exclusion of dependencies:

1. The dependency FIA_UID.1 of FMT_SMR.1/Installer is discarded. The Java Card PP [PP-JC] does not require the identification of the "installer" since it can be considered as part of the TSF.
2. The dependency FIA_UID.1 of FMT_SMR.1/ADEL is discarded. The Java Card PP [PP-JC] does not require the identification of the "deletion manager" since it can be considered as part of the TSF.
3. The dependency FMT_SMF.1 of FMT_MSA.1/JCRE is discarded. The dependency between FMT_MSA.1/JCRE and FMT_SMF.1 is not satisfied because no management functions are required for the Java Card RE.
4. The dependency FAU_SAA.1 of FAU_ARP.1 is discarded. The dependency of FAU_ARP.1 on FAU_SAA.1 assumes that a "potential security violation" generates an audit event. On the contrary, the events listed in FAU_ARP.1 are self-contained (arithmetic exception, ill-formed bytecodes, access failure) and ask for a straightforward reaction of the TSFs on their occurrence at runtime. The JCVM or other components of the TOE detect these events during their usual working order. Thus, there is no mandatory audit recording in the Java Card PP [PP-JC].

9.3.2 Firmware Upgrade OS

SFR	Dependency	Satisfied by
FTP_ITC.1/Loader	None.	n/a
FDP_ACC.1/Loader	FDP_ACF.1 Security attribute based access control	FDP_ACF.1/Loader
FDP_ACF.1/Loader	FMT_MSA.3 Static attribute initialization FDP_ACC.1 Subset access control	FMT_MSA.3 is not required because the security attributes used to enforce the Loader SFP are fixed during manufacturing phase and no new objects under control of the Loader SFP are created. FDP_ACC.1/Loader

9.4 **Rationale for the Security Assurance Requirements**

EAL6 is required for this type of TOE and product since it is intended to defend against sophisticated attacks. This evaluation assurance level allows a developer to gain maximum assurance from positive security engineering based on good practices. EAL6 represents the highest practical level of assurance expected for a commercial grade product. In order to provide a meaningful level of assurance that the TOE and its embedding product provide an high level of defense against such attacks: the evaluators should have access to the low level design and source code.

9.4.1 ALC_FLR.2 Flaw reporting procedures

ALC_FLR.2 add additional assurance to EAL6
Dependencies: No dependencies.
Objectives: in order for the developer to be able to act appropriately upon security flaw reports from TOE users, and to know to whom to send corrective fixes, TOE users need to understand how to submit security flaw reports to the developer.

Flaw remediation guidance from the developer to the TOE user ensures that TOE users are aware of this important information.

9.5 IC Composition rationale

9.5.1 Common Criteria rationale

Assurance level of the IC evaluation is EAL6 augmented by ALC_FLR.2.

Assurance level of the current evaluation is consistent with the assurance level in IC evaluation.

9.5.2 Compatibility between threats (TOE and IC)

IC Threats	Rationale
BSI.T.Leak-Inherent	This threat is related to the information which is leaked from the TOE during usage of the Security IC in order to disclose sensitive data of the TOE. This threat has been considered in the current evaluation.
BSI.T.Phys-Probing	This threat is related to physical probing of the TOE to disclose relevant information. This threat has been considered in the current evaluation.
BSI.T.Malfunction	This threat is related to force malfunctions of the TSF due to environmental stress that could lower or bypass the implemented security mechanisms. This threat has been considered in the current evaluation.
BSI.T.PhysManipulation	This threat is related to physical manipulation of the Security IC. This is covered by the IC evaluation.
BSI.T.Leak-Forced	This threat is related to information which is leaked from the TOE during usage of the Security IC in order to disclose confidential user data of the composite TOE. This is covered by the IC evaluation.
BSI.T.Abuse-Func	This threat is related to the usage of functions of the TOE that are not allowed once the TOE Delivery and can impact the security of the TOE. This threat has been considered in the current evaluation.
BSI.T.RND	This threat is related to the deficiency of random numbers. This is covered by the IC evaluation.
BSI.T.Masquerade-TOE	This threat relates to the capacity of an attacker to produce a chip which is not a genuine TOE but wrongly identifying itself as genuine TOE sample. Mitigation of masquerade requires tightening up the identification by authentication and is covered by IC evaluation.
AUG4.T.Mem-Access	The TOE implements memory access violation mechanisms based on the IC security policy. Therefore, this threat also covered by the TOE evaluation.
JIL.T.Open-SamplesDiffusion	This threat refers to the diffusion of open samples: an attacker may get access to open samples of the TOE and use them to gain information about the TSF (loader, memory management unit, ROM code, …). This threat is also covered by the TOE evaluation.

9.5.3 Compatibility between assumptions (TOE and IC)

IC Assumptions	Rationale
BSI.A.Process-Sec-IC	This assumption ensures the security of the delivery and storage of the IC. It is covered by the ALC_DVS.2 activity of the current TOE evaluation.
BSI.A.Resp-Appl	This assumption ensures that security relevant data of the current TOE are properly treated according to the IC security needs. It is covered by the ADV_IMP.2 activity of the TOE evaluation.

9.5.4 Compatibility between security objectives for the environment (TOE and IC)

IC OEs	Rationale
BSI.OE.Resp-Appl	This objective deals with the treatment of TOE user data by the TOE itself. It is covered by the ADV_IMP activity of the TOE evaluation.
BSI.OE.Process-Sec-IC	This objective is covered by the IC evaluation.
BSI.OE.Lim-Block-Loader	This objective is covered by the IC evaluation.
BSI.OE.Loader-Usage	This objective is covered by the IC evaluation.
BSI.OE.TOE-Auth	This objective is covered by the IC evaluation.
OE.Composite-TOE-Id	Also covered by the current evaluation.
OE.TOE-Id	This objective is covered by the IC evaluation.
OE.Enable-Disable-Secure-Diag	Secure Diagnosis feature can be enabled/disabled by the composite TOE manufacturer, so the OE is also covered by the current evaluation.
OE.Secure-Diag-Usage	This objective is covered by the IC evaluation.

9.5.5 Compatibility between Security Objectives (TOE and IC)

BSI.O.Leak-Inherent	Also covered by the current evaluation.
BSI.O.Phys-Probing	Also covered by the current evaluation.
BSI.O.Malfunction	Also covered by the current evaluation.
BSI.O.Phys-Manipulation	Covered by the IC evaluation.
BSI.O.Leak-Forced	Covered by the IC evaluation.
BSI.O.Abuse-Func	Also covered by the current evaluation.
BSI.O.Identification	Covered by the IC evaluation.
BSI.O.RND	Covered by the IC evaluation.
BSI.O.Cap-Avail-Loader	Covered by the IC evaluation.
BSI.O.Ctrl-Auth-Loader	Also covered by the current evaluation.
BSI.O.Authentication	Also covered by the current evaluation.
JIL.O.Prot-TSF-Confidentiality	Covered by the IC evaluation.
JIL.O.Secure-Load-ACode	Covered by the IC evaluation.
JIL.O.Secure-AC-Activation	Covered by the IC evaluation.
JIL.O.TOE-Identification	Covered by the IC evaluation.
O.Secure-Load-AMemImage	Covered by the IC evaluation.
O.MemImage-Identification	Covered by the IC evaluation.
AUG1.O.Add-Functions	Covered by the IC evaluation.
AUG4.O.Mem-Access	Also covered by the current evaluation.

9.5.6 Compatibility between Organisational Security Policies (TOE and IC)

IC Policies	Rationale
BSI.P.Process-TOE	This policy is related to the accurate unique identification during IC Development and Production. It was covered by the IC evaluation.

BSI.P.Lim-Block-Loader	Limiting and blocking the loader functionality for loading of Security IC Embedded Software. It was covered by the ALC_DVS.2 activity of the current TOE evaluation.
BSI.P.Ctrl-Loader	Authorized user controls the usage of the Loader functionality in order to protect stored and loaded user data from disclosure and manipulation. It was covered by the IC evaluation.
AUG1.P.Add-Functions	Additional Specific Security Functionality is provided by the IC. It was covered by the IC evaluation.

9.5.7 Compatibility between SFRs (TOE and IC)

IC SFRs are separated in the following groups as defined in [SOGIS-COMP]:

1. IP_SFR: irrelevant IC SFR not being used by the current TOE.
2. RP_SFR-SERV: relevant IC SFR being used by the current TOE to implement a security service with associated TSFI.
3. RP_SFR-MECH: relevant IC SFR being used by the current evaluation because its security properties providing protection attacks to the TOE.

IC SFR	**Rationale**
FRU_FLT.2	RP_SFR-MECH
FPT_FLS.1	RP_SFR-MECH
FMT_LIM.1/Test	RP_SFR-MECH
FMT_LIM.2/Test	RP_SFR-MECH
FMT_LIM.1/Loader	RP_SFR-MECH
FMT_LIM.2/Loader	RP_SFR-MECH
FMT_LIM.1/Stest	IP_SFR
FMT_LIM.2/Stest	IP_SFR
FMT_LIM.1/Sdiag	RP_SFR-SRV
FMT_LIM.2/Sdiag	RP_SFR-SRV
FAU_SAS.1	RP_SFR-MECH
FDP_SDC.1	RP_SFR-MECH
FDP_SDI.2	RP_SFR-MECH
FPT_PHP.3	RP_SFR-MECH
FDP_ITT.1	RP_SFR-MECH
FPT_ITT.1	RP_SFR-MECH
FDP_IFC.1	RP_SFR-MECH
FCS_RNG.1/PTG.2	RP_SFR_SERV
FCS_COP.1	RP_SFR_SERV
FDP_ACC.2/Memories	RP_SFR-MECH
FDP_ACF.1/Memories	RP_SFR-MECH
FMT_MSA.3/Memories	RP_SFR-MECH
FMT_MSA.1/Memories	RP_SFR-MECH
FMT_SMF.1/Memories	RP_SFR-MECH
FIA_API.1	RP_SFR-MECH
FDP_ITC.1/Loader	RP_SFR-SERV
FDP_UCT.1/Loader	RP_SFR-SERV
FDP_UIT.1/Loader	RP_SFR-SERV
FDP_ACC.1/Loader	RP_SFR-SERV
FDP_ACF.1/Loader	RP_SFR-SERV

FMT_MSA.3/Loader	RP_SFR-SERV
FMT_MSA.1/Loader	RP_SFR-SERV
FMT_SMR.1/Loader	RP_SFR-SERV
FIA_UID.1/Loader	RP_SFR-SERV
FIA_UAU.1/Loader	RP_SFR-SERV
FMT_SMF.1/Loader	RP_SFR-SERV
FPT_FLS.1/Loader	RP_SFR-SERV
FAU_SAS.1/Loader	RP_SFR-SERV
FAU_SAR.1/Loader	RP_SFR-SERV
FTP_ITC.1/Sdiag	RP_SFR-SERV
FAU_SAR.1/Sdiag	RP_SFR-SERV

10 Abbreviations and glossary

[CC]	Common Criteria
[EAL]	Evaluation Assurance Level
[LPU]	Library Protection Unit
[MPU]	Memory Protection Unit
[NVM]	Non-Volatile Memory
[ST]	Security Target
[TOE]	Target of Evaluation
[TSF]	TOE Security Functionality
[PP]	Protection Profile

11 References

[ANSI X9.31]	Digital Signatures using Reversible Public Key Cryptography for the Financial Services Industry (rDSA), ANSI X9.31-1998, American Bankers Association
[AGD_OPE]	STeID JC Open OS v1.0 Operational Guidance (AGD_OPE), rev. 3
[AGD_PRE]	STeID JC Open OS v1.0 Preparative Procedure (AGD_PRE), rev. 3
[AIS20/31]	Bundesamt fuer Sicherheit in der Informationstechnik. AIS20/31: A proposal for: Functionality classes for random number generators, Bundesamt für Sicherheit in der Informationstechnik (BSI), Version 2.0, September 18th, 2011.
[CERT-IC]	ST31N600 A03, ANSSI - CC - 2022/21 – R02
[CC31R5P1]	Common Criteria for Information Technology Security Evaluation. Part 1: Introduction to General Model, Version 3.1, Revision 5, April 2016.
[CC31R5P2]	Common Criteria for Information Technology Security Evaluation. Part 2: Security functional components, Version 3.1, Revision 5, April 2016.
[CC31R5P3]	Common Criteria for Information Technology Security Evaluation. Part 3: Security assurance components, Version 3.1, Revision 5, April 2016.
[FIPS 46-3]	FIPS PUB 46-3, Data Encryption Standard, October 25, 1999 (ANSI X3.92), National Institute of Standards and Technology
[FIPS 81]	FIPS PUB 81, DES Modes of Operation, April 17, 1995, National Institute of Standards and Technology
[FIPS 140-2]	FIPS PUB 140-2, Security requirements for cryptographic modules, March 2002 , National Institute of Standards and Technology
[FIPS 180-2]	FIPS PUB 180-2 Secure Hash Standard with Change Notice 1 dated February 25,2004, National Institute of Standards and Technology, U.S.A., 2004
[FIPS 197]	FIPS PUB 197, The Advanced Encryption Standard (AES) U.S. DoC/NIST, November 26, 2001
[GP]	Global Platform Inc., Global Platform Card Specification 2.3, October 2015. Document Reference: GPC_SPE_034
	GlobalPlatform Card - Composition Model - Security Guidelines for Basic Applications - Version 1.0 - June 2012 – ref. GPC_GUI_050
[GP-D]	GlobalPlatform Inc., GlobalPlatform Card Technology, Secure Channel Protocol ‘03’, Card Specification v2.2 – Amendment D, Version 1.1.1, July 2014
[GP-H]	GlobalPlatform Inc., GlobalPlatform Technology. Executable Load File Upgrade Card Specification v2.3 - Amendment H, Version 1.1, March 2018
[GP-SGBA]	GlobalPlatform Card - Composition Model - Security Guidelines for Basic Applications - Version 1.0 - June 2012 – ref. GPC_GUI_050
[IEEE 1363a]	IEEE Std 1363a-2004 Standard Specification of Public-Key Cryptography
[JCVM]	Java Card Virtual Machine Java Card Platform, Version 3.0.5, 2015, Oracle Technology Network
[JCAPI]	Java Card Application Programming Interfaces, Version 3.0.5, 2015, Oracle Technology Network
[JCRE]	Java Card Runtime Environment Specification, Classic Edition Version 3.0.5, 2015, Oracle Technology Network
[KS2011]	W. Killmann, W. Schindler, „A proposal for: Functionality classes for random number generators“, Version 2.0, September 18, 2011
[PP-IC]	Security IC Platform Protection Profile with Augmentation Packages Version 1.0 - BSI-CC-PP-0084-2014
[PP-JC]	Java Card System - Open Configuration Protection Profile, April 2020, Version 3.1

[SOGIS-COMP] Composite product evaluation for Smart Cards and similar devices, version 1.5.1, May 2018

[NISTSP800-90] NIST SP 800-90 NIST Special Publication 800-90, Recommendation for random number generation using deterministic random bit generators (Revised), National Institute of Standards and Technology (NIST), March 2007

[ICAO] ICAO Doc 9303, Machine Readable Travel Documents – Part 11 Security Mechanisms for MRTDs – Eighth Edition, 2021