SECORA™ ID S v1.1 (SLJ52GxxyyyzS)

Security Target

About this document

Scope and purpose

This document contains the Security Target for the evaluation of the **SECORA™ ID S v1.1 (SLJ52GxxyyyzS)** Java Card according to Common Criteria EAL6 augmented with ALC_FLR.1.

Intended audience

Common Criteria evaluators, Common Criteria certification bodies, Composite product (applet) developers

Table of Contents

1 Introduction

1.1 ST reference

Title: SECORA™ ID S v1.1 (SLJ52GxxyyyzS)

Version: Rev 1.9

Publication date: 2021-01-27

Sponsor: Infineon Technologies AG, 81726 Munich, Germany

Editor: Infineon Technologies AG, 81726 Munich, Germany

1.2 TOE reference

TOE Name: SECORA™ ID S v1.1 (SLJ52GxxyyyzS)

The TOE consists of several variants which are reflected in the TOE name. The letters x, y, and z are placeholders for the following values:

- The first variable x is for the available interface (can be ‘C’, ‘L’, or ‘D’ for the contact based, contactless or dual Interface)
- The second variable x is for the available RSA cryptography library (‘T’ stands for 2K RSA , ‘A’ for 4K RSA)
- The 3 digit variable yyy is the available user memory in kB
- The variable z is a place holder for products that will be based on the TOE (e.g. ‘A’ for ePassport with HBR, 'B’ for eDriving License with HBR, ‘C’ for National eID open platform configuration with HBR, ‘D’ for National eID with applications and HBR or ‘V’ for open platform configuration with VHBR, etc.)

CC certificate number: NSCIB-CC-175887_2

1.2.1 Underlying hardware platform

CC certificate number: BSI-DSZ-CC-1110-V3-2020

CC Identifier: IFX_CCI_000005

Security Target: [ST_IC]

1.2.2 Dedicated software

1.3 TOE overview

The TOE is a Java Card Platform compliant with Java Card Specification (Classic Edition) version 3.0.5 ([JCAPI3], [JCRE3], [JCVM3]) and GlobalPlatform Specification v.2.3.1 ([GP v23], [GPv23 Amd D]) and the GlobalPlatform Card ID Configuration v1.0 [GP-ID]. The TOE allows post-issuance downloading of applications that have been previously verified by an off-card verifier. It constitutes a secure generic platform that supports multi-application runtime environment and provides facilities for secure loading and interoperability between different applications.

1.3.1 TOE type

The TOE is a Java Card with a GP Framework. It can be used to load and execute off-card verified Java Card applets.

1.3.2 TOE features

1.3.2.1 Java Card OS standard features

Java Card open platform

The Java Card OS supports an open platform mode. In this mode loading, installation and deletion of several applet packages are permitted post issuance. This is default mode.

Cryptographic ciphers

The Java Card OS supports the following cryptographic algorithms:

- AES 128/192/256 Cipher Scheme for secure messaging (ENC), message authentication (MAC) and authentication procedures
- TDES Cipher Scheme for secure messaging (ENC), message authentication (MAC) and authentication procedures.
- RSA encryption and decryption up to 4k

Signature algorithms

- ECDSA with SHA-1/SHA-2
- RSA PKCS#1 with SHA-2
- RSA PSS with SHA256

Key agreement algorithms

- ECDH with KDF and with XY
- PACE with generic mapping

Key pair generation

- EC
- RSA with modulus/exponent and CRT

Key Sizes

- AES 128/192/256
- TDES 128/192
- RSA modulus sizes from 512 to 4096 bits
- EC curves according to NIST and Brainpool
  - NIST standard curves from FIPS 186-3: P224, P256, P384, P521
  - Brainpool curves from RFC 5639: BrainpoolP224, BrainpoolP256r1, BrainpoolP320r1, BrainpoolP384r1,BrainpoolP512r1, BrainpoolP256t1,BrainpoolP320t1,BrainpoolP384t1, BrainpoolP512t1

Message digest algorithms

- SHA-1
- SHA-2 family: SHA224, SHA256, SHA384, SHA512

Note: SHA-1 as a security algorithm is only used as part of a session key derivation.

Random number generation algorithms

- Hybrid physical RNG according to [AIS 31] PTG.3

1.3.2.2 Java Card OS proprietary features

Java Card static mode

The Java Card OS supports a proprietary “Java Card Static” mode. The OS provides an APDU command to activate the static mode. In this mode the installation of new applet instances (based on the packages already present on the card) is allowed. Deletion of already present applet instances and applet packages is allowed. Loading of additional packages is not allowed.

Java Card native mode

The Java Card OS supports a proprietary “Java Card native” mode. The OS provides an APDU command to activate the native mode. In this mode, full privacy is provided by removing any means to get tracking information. In particular, no GP functions and no TOE identification commands are available.

LDS-API

The Java Card OS supports a proprietary accelerated API for ICAO DOC 9303 ([DOC9303]) compliant secure messaging and session key derivation.

PACE API

The Java Card OS supports a proprietary API for the PACE cryptographic protocol which is especially hardened againts side channel attacks. This ensures that PACE can be used with low entropy PINs

1.3.2.3 GlobalPlatform features

GlobalPlatform ID profile

The TOE is compliant to the GP v2.3.1 [GP v23] with GP ID profile V1.0 [GP-ID].

Secure channel

The TOE supports the following secure channel protocols

- SCP 02 with option 15 and 55 according to [GP v23]
- SCP 03 with option 10 and 00 according to [GPv23 Amd D]

Logical channel

The TOE supports 4 logical channels

Optional APIs

- org.globalplatform.Personalization

Global object

The TOE supports a global PIN CVM application, implementing velocity checking. The CVM value can be in any of the formats BCD, ASCII, HEX.

TOE identification

The TOE provides a proprietary function to return TOE identification. This command is disabled in case the TOE is in native mode.

Administration options

The TOE supports multiple security domains to allow independent entities administrate their assigned applications.

- Deletion of applications via Authorized Management (AM) and Delegated Management (DM).
- Installation of applications via AM and DM
- Loading Executable Load Files via AM and DM
- Removal of packages via AM and DM
- Generate receipts for successfully executed Delegated Management Operations based on TDES & AES.
- If configured, the Security Domain verifies the Delegated Management Tokens based on RSA before executing card management operations.
- Extradition of application instances via AM and DM
- Extradition of packages via AM and DM
- The TOE supports DAP for single or multiple blocks with asymmetric keys based on RSA 2k

1.3.3 Non TOE HW/SW/FW available to the TOE

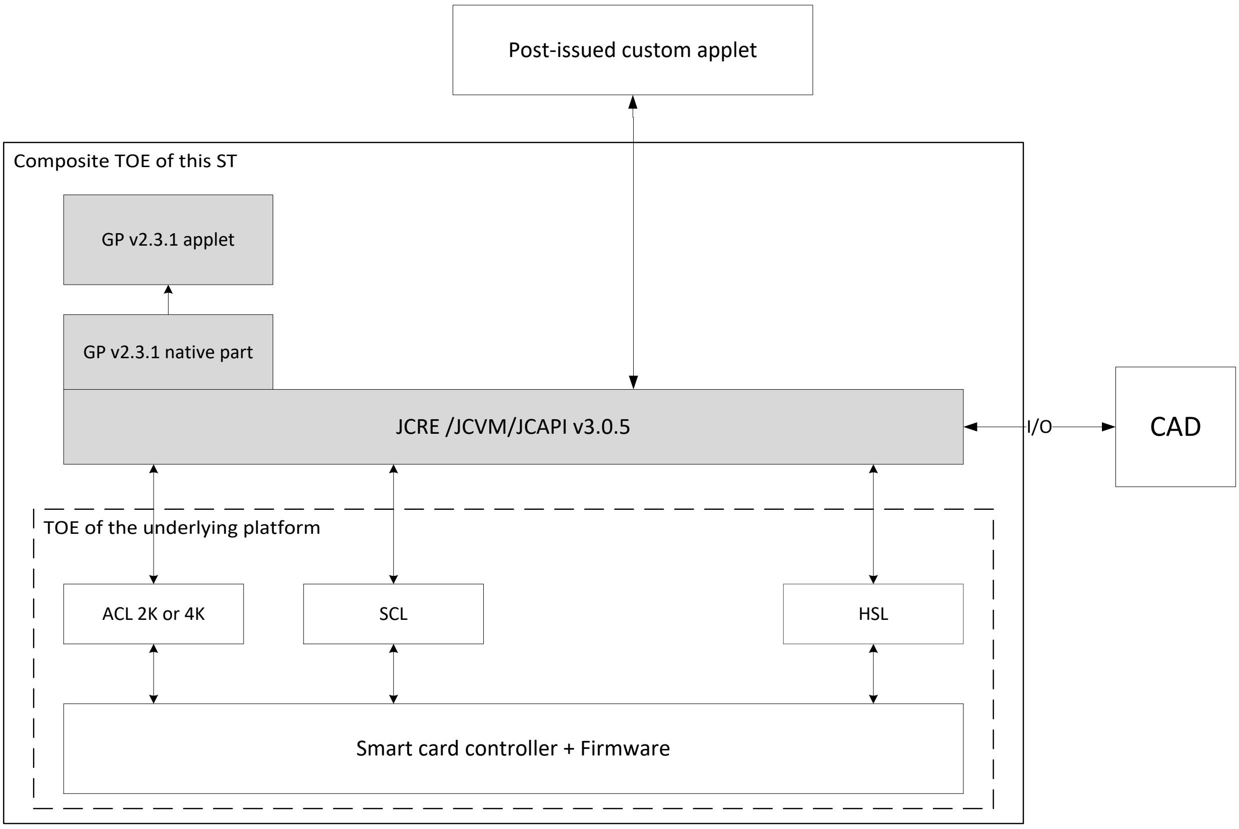
Bytecode verification must be done off-card with the latest version of the bytecode verifier.

1.4 TOE description

1.4.1 TOE components

The figure below shows the composite TOE. The evaluation focus of this ST is the gray filled part.

Figure 1 TOE overview



1.4.1.1 Underlying platform

The underlying platform is a certified smart card controller with Common Criteria certificate number BSI-DSZ-CC-1110-V3-2020 with identifier IFX_CCI_000005. The smart card controller can be delivered with three different default input capacitances of the contactless antenna pads, 27pf, 56pf or 78pf. For details, please see [ST_IC].

1.4.1.2 IC dedicated software

The firmware of the underlying platform has version 80.100.17.3.

The following dedicated software libaries are part of the underlying platform certification. Those libraries will not be delivered to the customers of this TOE.

- **Asymmetric Cryptographic Library(ACL)**: This library contains hardened code for RSA and ECC cryptography. The RSA library consists in 2 variants, the 2K version which supports RSA up to 2048 bits and the 4K version which supports RSA up to 4096 bits. The used RSA library is part of the configuration options. The library supports ECC up to 521 bits. The ACL also contains the Toolbox API, which provides basic arithmetical primitives.
- **Symmetric Cryptographic Library(SCL)**: This library contains hardened code for AES and TDES cryptography
- **Hardware Support Library(HSL)**: This library contains code for Non-Volatile Memory (NVM) read/write access

No other optional IC dedicated software library is used in the embedded software.

1.4.1.3 IC embedded software

The IC embedded software consists of JCVM 3.0.5 [JCVM3], JCRE 3.0.5 [JCRE3] , JCAPI 3.0.5 [JCAPI3] and GP 2.3.1 ([GP v23], [GPv23 Amd D]) framework with ID configuration 1.0 [GP-ID]. Additionally it consists of a proprietary API which is described in [DBOOK].

1.4.1.4 Guidance documentation

The TOE will be delivered with the following guidance documentation

Table 1 Guidance documentation

Document	Version	Date
SECORA™ ID S v1.1 Administrator Guide	1.70	2021-01-27
SECORA™ ID S v1.1 Data Book	1.70	2021-01-27
SECORA™ ID S v1.1 Security Guidance	2.20	2021-01-27
SECORA™ ID S v1.1 SLJ52GxAyyyzS System Release Notes	2.20	2021-01-27
SECORA™ ID S v1.1 SLJ52GxTyyyzS System Release Notes	2.20	2021-01-27
SECORA™ ID S v1.1 Product API Specification	1.02.1442	2020-11-18

1.4.2 TOE identification

The TOE can be uniquely identified as described in [SECGUIDE].

The TOE belonging to this Security target is uniquely identified by the following data:

Table 2 TOE identification data

Component	Reference value
CC Identifier of underlying platform	IFX_CCI_000005
Embedded OS version	1442
ACL version	2.07.003
SCL version	2.04.002
HSL version	03.12.8812

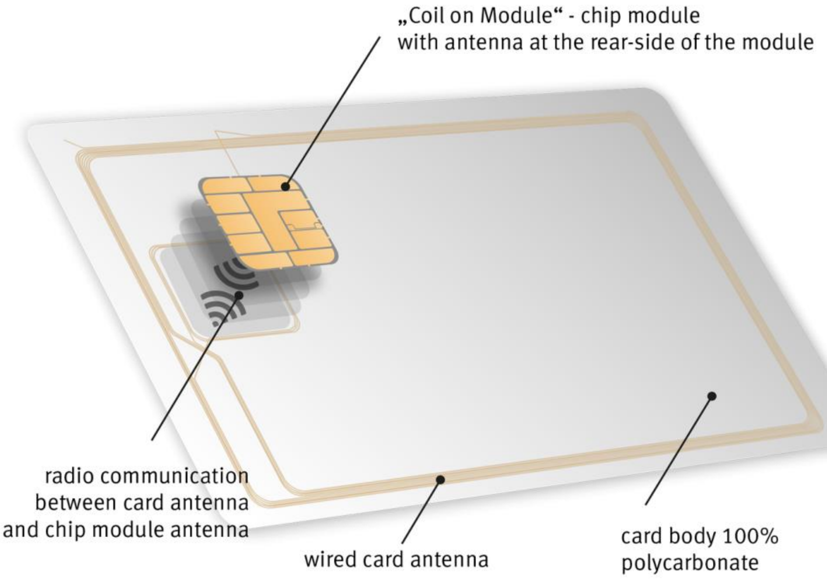
1.4.3 TOE package types

The TOE can be delivered in the following forms:

- Packaged as
  - contact based modules
  - dual interface modules
  - contactless modules
- Packageless as sawn or unsawn wafer

The TOE supports Coil on Module antennas for dual interface modules.

Figure 1 Coil on Module example

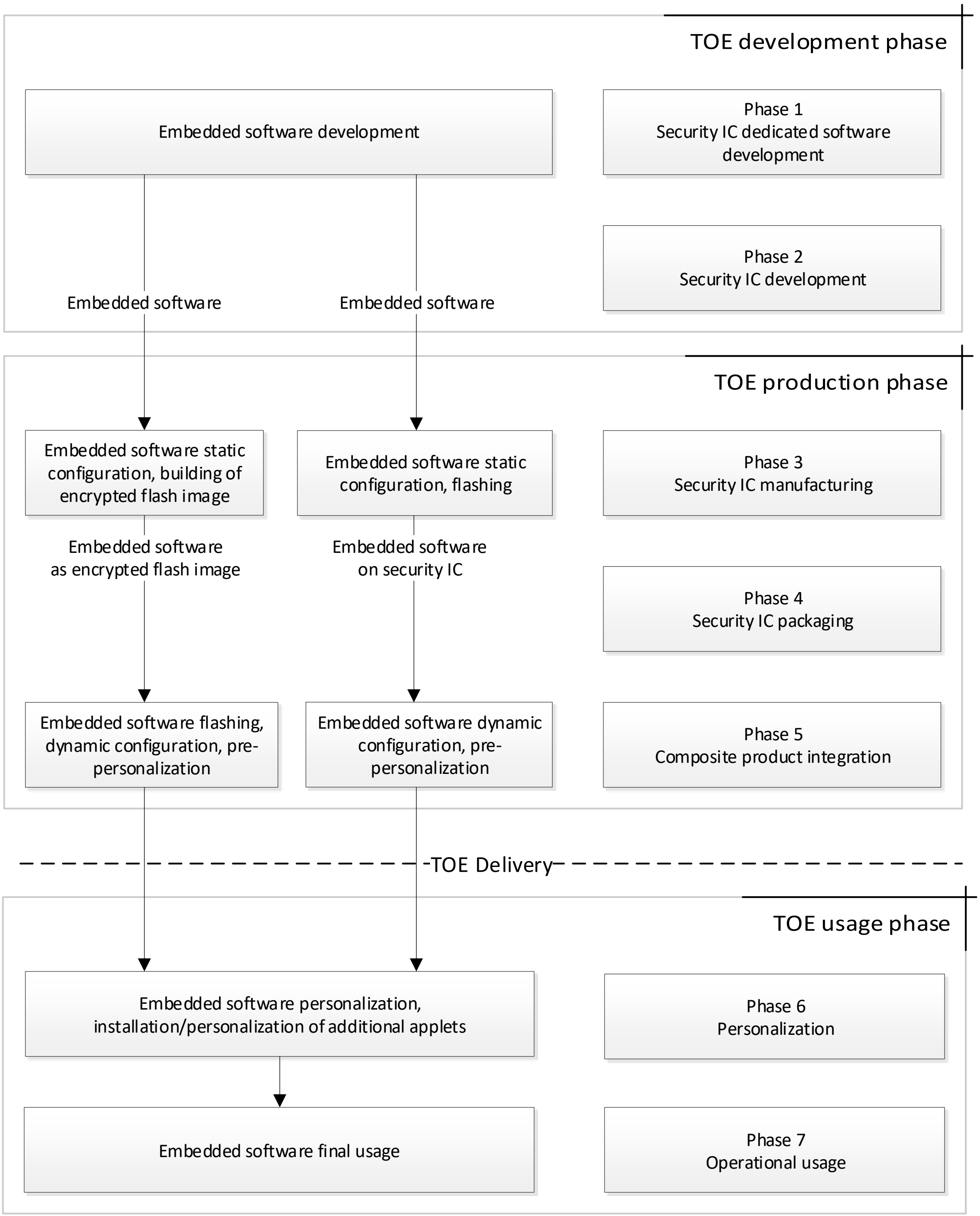


1.4.4 TOE life cycle

The life cycle of this composite TOE is associated to the life cycle of the underlying hardware platform. The association is illustrated in Figure 2. The delivery points are marked with double-arrows

A detailed description of the life cycles of the underlying hardware IC can be found in [ST_IC].

Figure 2 TOE life cycle


TOE development phase
Embedded software development
Phase 1
Security IC dedicated software development
Phase 2
Security IC development
Embedded software
Embedded software
TOE production phase
Embedded software static configuration, building of encrypted flash image
Embedded software static configuration, flashing
Phase 3
Security IC manufacturing
Embedded software as encrypted flash image
Embedded software on security IC
Phase 4
Security IC packaging
Embedded software flashing, dynamic configuration, pre-personalization
Embedded software dynamic configuration, pre-personalization
Phase 5
Composite product integration
TOE Delivery
TOE usage phase
Embedded software personalization, installation/personalization of additional applets
Phase 6
Personalization
Embedded software final usage
Phase 7
Operational usage

There are two life cycle variants of the composite TOE, depending on where the flashing of the embedded software takes place:

- The embedded software is flashed on the IC during phase 3 of the IC life cycle. This can only be done by Infineon.
- The embedded software is flashed on the IC during phase 5 of the IC life cycle. This can only be done by Infineon.

1.4.4.1 Flashing of embedded software in phase 3

This is the right hand path of the diagram in Figure 2.

- Phase 1: The embedded software development (i.e. Java Card OS, GlobalPlatform API applet and additional applets) takes place.
- Phase 3: Static configuration of the embedded software hex image is done. Pre-loaded applets and the GP ISD keys will be included in the hex image via the templating mechanism. Flashing of the hex image on the IC takes place.
- Phase 5: Dynamic configuration data can be written or changed via proprietary APDU commands. The user is Infineon or a third party composite product integrator. The TOE will be delivered to the customer.
- Phase 6: The applets will be personalized. Additional applets can be installed by using GlobalPlatform commands.
- Phase 7: The Java Card operating system and its applets are in operational phase.

1.4.4.2 Flashing of embedded software in phase 5

This is the left hand path of the diagram in Figure 2.

- Phase 1: The embedded software development (i.e. Java Card OS, GlobalPlatform API applet and additional applets) takes place.
- Phase 3: Static configuration of the embedded software hex image is done. Pre-loaded applets can be included in the hex image via the templating mechanism. Transport keys for GP ISD are written in the hex image. The hex image for the embedded software will be encrypted with the key of the composite product integrator. The user is always Infineon.
- Phase 5: Applets can be installed via GlobalPlatform APDU commands. Configuration data can be written or changed via proprietary APDU commands. The user is a third party composite product integrator.
- Phase 6: The applets will be personalized. Additional applets can be installed by using GlobalPlatform commands. After that phase, the Java Card with its applet will be handed over to the end customer.
- Phase 7: The Java Card operating system and its applets are in operational phase.

1.4.4.3 Templating mechanism

Templating is a production method, where installation and pre-personalisation of applets can be done in phase 3 of the Java Card life cycle.

Templating is done with a special templating card (special configurator option) or on a card in a special life-cycle state or a card simulator. Only one option needs to be implemented. On this card or simulation,

- the operating system can be initialized
- applets can be loaded and/or installed
- GlobalPlatform keys can be loaded or exchanged
- Applications can be pre-personalized with common data

After those steps, a “NVM Image” of the chip can be extracted.

The NVM Image will be sent to an Infineon production site.
NVM Image creation and NVM Image loading is always performed in a secure and certified environment.

1.4.5 TOE configurations

The TOE can be delivered with different configuration options. The following table shows all possible configuration options and associated life cycles, where the changes are permitted.

Changes in lifecycle phase 3 are done by directly patching the binary image or by sending APDU commands to a simulator image. It can only be done by the manufacturer Infineon. The RSA 2K and 4K options are not selected by patching but are different binary images.

Changes in lifecycle phase 6 are done by specific APDU commands (see [ADMIN]). It can be done by the personalizer or composite product developer.

Table 3 TOE configurations

Configuration item	Life cycle
RSA 2K or 4K	1
Configurable Heap Size in 1kB granularity for applet instances and applet data.	3
APDU Buffer Size (APDU including header) in 261 (default), 512, 1024 and 1500 byte.	3
CPLC data	3,6
ATR	3,6
Deactive CIM (default) or active CIM	3
Random and fixed PUPI/UID	3,6
Contactbased: T=0, T=1,divider = 16 @ 5 MHz	3,6
Contactless: Type A or B	3,6
106, 212, 424, 848 kbaud/s with asymmetric baudrates; VHBR up to 6.8 Mb/s (on/off), Higher Frame Size up to 1024 bit (on/off)	3
FWI values in the range of 6 to 14	3,6
ATS or ATTRIB	3,6
Blocking of Applet-Loading (deactivate command INSTALL [for load]) by APDU-command	6
Blocking of Applet-Loading (deactivate command INSTALL [for load]) by configuration tool (static, modifying HEX-file)	3
Blocking of Applet-Loading (deactivate command INSTALL [for load]) by configuration tool (static, modifying HEX-file) autonomously when GP state SECURED is reached	3

1.4.6 Forms of delivery

Table 4 TOE deliveries: forms and methods

TOE Component	Delivered Format	Delivery Method	Comment
Underlying platform with Java Card OS	See ch. 1.4.3	Postal transfer in cages	All materials are delivered to distribution centers in cages, locked.
All User Guidance documents	Personalized PDF	SecureX transfer	

2 Conformance claims

2.1 CC Conformance Claims

Conformance of this security target is claimed for CC part 2 revision 5 [CC2] extended by FCS_RNG.1 and CC part 3 revision 5 [CC3] conformant.

The Evaluation Assurance Level is EAL 6 augmented by ALC_FLR.1.

2.2 Protection profile conformance claims

This Security Target is in demonstrable conformance to the Java Card Protection Profile Open Configuration Version 3.0 [JC PP].

The following adaptions with respect to the [JC PP] have been done for this ST

- The [JC PP] refers to Java Card version 3.0.1, but this ST uses version 3.0.5
- The [JC PP] refers to GP version 2.2, but, this ST uses version 2.3.1
- The [JC PP] refers to the [PP0035] protection profile for the hardware IC, but this ST (and also its underlying platform) refers to [PP0084], which is a superset of [PP0035].

2.2.1 Demonstrable conformance claim rationale

- The following security objectives for the operational environment and assumptions have been mapped to security objectives for the TOE in this ST:

Table 5 Mapping of OE to Objectives

OE objectives of underlying platform	TOE objectives
OE.CARD-MANAGEMENT	O.CARD-MANAGEMENT
OE.SCP.IC	O.SCP.IC
OE.SCP.RECOVERY	O.SCP.RECOVERY
OE.SCP.SUPPORT	O.SCP.SUPPORT

-
- Due to the introduction of O.CARD-MANAGEMENT, the assumption A.DELETION has been removed.
- The objective O.CARD-MANAGEMENT has been refined to include GlobalPlatform specific objectives.
- The following threats have been added:

Table 6 Additional threats and associated objectives

Threats	TOE Objectives
T.COMMUNICATION	O.COMMUNICATION
T.RNG	O.SCP.RNG
T.CARD-MANAGEMENT	O.CARD-MANAGEMENT

- A new SFR group CMCGR has been introduced to reflect the integration of the GlobalPlatform card and content management. The SFRs from the group CMCGR have been taken from [JC USIM PP] but conformity to this PP is not claimed. Therefore, all SFRs of CMCGR shall be considered as defined in this ST.
- The group CarG has been refined by the group CMCGR. An SFR mapping from the CarG to the CMCGR group is given in chapter 7.2.5.

- The JCRMI and EMG group are not part of the TOE and have been removed for this ST.

3 Security aspects

This chapter describes the main security issues of the Java Card System and its environment addressed in this Security Target, called "security aspects", in a CC-independent way.

In addition to this, they also give a semi-formal framework to express the CC security environment and objectives of the TOE. They can be instantiated as assumptions, threats, objectives (for the TOE and the environment) or organizational security policies. For instance, we will define hereafter the following aspect:

#.OPERATE (1) The TOE must ensure continued correct operation of its security functions. (2) The TOE must also return to a well-defined valid state before a service request in case of failure during its operation.

TSFs must be continuously active in one way or another; this is called "OPERATE". The Security Target may include an assumption, called "A.OPERATE", stating that it is assumed that the TOE ensures continued correct operation of its security functions, and so on. However, it may also include a threat, called "T.OPERATE", to be interpreted as the negation of the statement #.OPERATE. In this example, this amounts to stating that an attacker may try to circumvent some specific TSF by temporarily shutting it down. The use of "OPERATE" is intended to ease the understanding of this document.

This section presents security aspects that will be used in the remainder of this document. Some being quite general, we give further details, which are numbered for easier cross-reference within the document. For instance, the two parts of #.OPERATE, when instantiated with an objective "O.OPERATE", may be met by separate SFRs in the rationale. The numbering then adds further details on the relationship between the objective and those SFRs.

All security aspects in this chapter have been taken verbatim from [JC PP].

Table 7 Confidentiality

#.CONFID-APPLI-DATA	Application data must be protected against unauthorized disclosure. This concerns logical attacks at runtime in order to gain read access to other application's data.
#.CONFID-JCS-CODE	Java Card System code must be protected against unauthorized disclosure. Knowledge of the Java Card System code may allow bypassing the TSF. This concerns logical attacks at runtime in order to gain a read access to executable code, typically by executing an application that tries to read the memory area where a piece of Java Card System code is stored.
#.CONFID-JCS-DATA	Java Card System data must be protected against unauthorized disclosure. This concerns logical attacks at runtime in order to gain a read access to Java Card System data. Java Card System data includes the data managed by the Java Card RE, the Java Card VM and the internal data of Java Card platform API classes as well.

Table 8 Integrity

#.INTEG-APPLI-CODE	Application code must be protected against unauthorized modification. This concerns logical attacks at runtime in order to gain write access to the memory zone where executable code is stored. In post-issuance application loading, this threat also concerns the modification of application code in transit to the card.
#.INTEG-APPLI-DATA	Application data must be protected against unauthorized modification. This concerns logical attacks at runtime in order to gain unauthorized write access

	to application data. In post-issuance application loading, this threat also concerns the modification of application data contained in a package in transit to the card. For instance, a package contains the values to be used for initializing the static fields of the package.
#.INTEG-JCS-CODE	Java Card System code must be protected against unauthorized modification. This concerns logical attacks at runtime in order to gain write access to executable code.
#.INTEG-JCS-DATA	Java Card System data must be protected against unauthorized modification. This concerns logical attacks at runtime in order to gain write access to Java Card System data. Java Card System data includes the data managed by the Java Card RE, the Java Card VM and the internal data of Java Card API classes as well.

Table 9 Unauthorized execution

#.EXE-APPLI-CODE	Application (byte)code must be protected against unauthorized execution. This concerns (1) invoking a method outside the scope of the accessibility rules provided by the access modifiers of the Java programming language ([JAVASPEC], §6.6); (2) jumping inside a method fragment or interpreting the contents of a data memory area as if it was executable code; (3) unauthorized execution of a remote method from the CAD.
#.EXE-JCS-CODE	Java Card System bytecode must be protected against unauthorized execution. Java Card System bytecode includes any code of the Java Card RE or API. This concerns (1) invoking a method outside the scope of the accessibility rules provided by the access modifiers of the Java programming language ([JAVASPEC], §6.6); (2) jumping inside a method fragment or interpreting the contents of a data memory area as if it was executable code. Note that execute access to native code of the Java Card System and applications is the concern of #.NATIVE.
#.FIREWALL	The Firewall shall ensure controlled sharing of class instances[7], and isolation of their data and code between packages (that is, controlled execution contexts) as well as between packages and the JCRE context. An applet shall not read, write, compare a piece of data belonging to an applet that is not in the same context, or execute one of the methods of an applet in another context without its authorization.
#.NATIVE	Because the execution of native code is outside of the JCS TSF scope, it must be secured so as to not provide ways to bypass the TSFs of the JCS. Loading of native code, which is as well outside those TSFs, is submitted to the same requirements. Should native software be privileged in this respect, exceptions to the policies must include a rationale for the new security framework they introduce.

Table 10 Bytecode verification

#.VERIFICATION	Bytecode must be verified prior to being executed. Bytecode verification includes (1) how well-formed CAP file is and the verification of the typing constraints on the bytecode, (2) binary compatibility with installed CAP files and the assurance that the export files used to check the CAP file correspond to those that will be present on the card when loading occurs.

Table 11 Card management

#.CARD-MANAGEMENT	(1) The card manager (CM) shall control the access to card management functions such as the installation, update or deletion of applets. (2) The card manager shall implement the card issuer's policy on the card.
#.INSTALL	(1) The TOE must be able to return to a safe and consistent state when the installation of a package or an applet fails or be cancelled (whatever the reasons). (2) Installing an applet must have no effect on the code and data of already installed applets. The installation procedure should not be used to bypass the TSFs. In short, it is an atomic operation, free of harmful effects on the state of the other applets. (3) The procedure of loading and installing a package shall ensure its integrity and authenticity.
#.SID	(1) Users and subjects of the TOE must be identified. (2) The identity of sensitive users and subjects associated with administrative and privileged roles must be particularly protected; this concerns the Java Card RE, the applets registered on the card, and especially the default applet and the currently selected applet (and all other active applets in Java Card System 2.2.x). A change of identity, especially standing for an administrative role (like an applet impersonating the Java Card RE), is a severe violation of the Security Functional Requirements (SFR). Selection controls the access to any data exchange between the TOE and the CAD and therefore, must be protected as well. The loading of a package or any exchange of data through the APDU buffer (which can be accessed by any applet) can lead to disclosure of keys, application code or data, and so on.
#OBJ-DELETION	(1) Deallocation of objects should not introduce security holes in the form of references pointing to memory zones that are not longer in use, or have been reused for other purposes. Deletion of collection of objects should not be maliciously used to circumvent the TSFs. (2) Erasure, if deemed successful, shall ensure that the deleted class instance is no longer accessible.
#DELETION	(1) Deletion of installed applets (or packages) should not introduce security holes in the form of broken references to garbage collected code or data, nor should they alter integrity or confidentiality of remaining applets. The deletion procedure should not be maliciously used to bypass the TSFs. (2) Erasure, if deemed successful, shall ensure that any data owned by the deleted applet is no longer accessible (shared objects shall either prevent deletion or be made inaccessible). A deleted applet cannot be selected or receive APDU commands. Package deletion shall make the code of the package no longer available for execution. (3) Power failure or other failures during the process shall be taken into account in the implementation so as to preserve the SFRs. This does not mandate, however, the process to be atomic. For instance, an interrupted deletion may result in the loss of user data, as long as it does not violate the SFRs. The deletion procedure and its characteristics (whether deletion is either physical or logical, what happens if the deleted application was the default applet, the order to be observed on the deletion steps) are implementation-dependent. The only commitment is that deletion shall not jeopardize the TOE (or its assets) in case of failure (such as power shortage). Deletion of a single applet instance and deletion of a whole package are functionally different operations and may obey different security rules. For instance, specific packages can be declared to be undeletable (for instance,

	the Java Card API packages), or the dependency between installed packages may forbid the deletion (like a package using super classes or super interfaces declared in another package).

Table 12 Services

#.ALARM	The TOE shall provide appropriate feedback upon detection of a potential security violation. This particularly concerns the type errors detected by the bytecode verifier, the security exceptions thrown by the Java Card VM, or any other security-related event occurring during the execution of a TSF.
#.OPERATE	(1) The TOE must ensure continued correct operation of its security functions. (2) In case of failure during its operation, the TOE must also return to a well-defined valid state before the next service request.
#.RESOURCES	The TOE controls the availability of resources for the applications and enforces quotas and limitations in order to prevent unauthorized denial of service or malfunction of the TSFs. This concerns both execution (dynamic memory allocation) and installation (static memory allocation) of applications and packages.
#.CIPHER	The TOE shall provide a means to the applications for ciphering sensitive data, for instance, through a programming interface to low-level, highly secure cryptographic services. In particular, those services must support cryptographic algorithms consistent with cryptographic usage policies and standards.
#.KEY-MNGT	The TOE shall provide a means to securely manage cryptographic keys. This includes: (1) Keys shall be generated in accordance with specified cryptographic key generation algorithms and specified cryptographic key sizes, (2) Keys must be distributed in accordance with specified cryptographic key distribution methods, (3) Keys must be initialized before being used, (4) Keys shall be destroyed in accordance with specified cryptographic key destruction methods.
#.PIN-MNGT	The TOE shall provide a means to securely manage PIN objects. This includes: (1) Atomic update of PIN value and try counter, (2) No rollback on the PIN-checking function, (3) Keeping the PIN value (once initialized) secret (for instance, no clear-PIN-reading function), (4) Enhanced protection of PIN's security attributes (state, try counter…) in confidentiality and integrity.
#.SCP	The smart card platform must be secure with respect to the SFRs. Then: (1) After a power loss, RF signal 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. (2) It does not allow the SFRs to be bypassed or altered and does not allow access to other low-level functions than those made available by the packages of the Java Card 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). (6) It safely transmits low–level exceptions to the TOE (arithmetic exceptions, checksum errors), when applicable. Finally, it is required that (7) the IC is designed in accordance with a well-defined set of policies and standards (for instance, those specified in [PP0035]), 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.
#.TRANSACTION	The TOE must provide a means to execute a set of operations atomically. This mechanism must not jeopardise the execution of the user applications. The transaction status at the beginning of an applet session must be closed (no pending updates).

4 Security problem definition

4.1 Assets

Assets are security-relevant elements to be directly protected by the TOE. Confidentiality of assets is always intended with respect to un-trusted people or software, as various parties are involved during the first stages of the smart card product life-cycle; details are given in threats hereafter.

Assets may overlap, in the sense that distinct assets may refer (partially or wholly) to the same piece of information or data. For example, a piece of software may be either a piece of source code (one asset) or a piece of compiled code (another asset), and may exist in various formats at different stages of its development (digital supports, printed paper). This separation is motivated by the fact that a threat may concern one form at one stage, but be meaningless for another form at another stage.

The assets to be protected by the TOE are listed below. They are grouped according to whether it is data created by and for the user (User data) or data created by and for the TOE (TSF data). For each asset it is specified the kind of dangers that weigh on it.

4.1.1 User data

Table 13 User data defined in [JC PP]

Asset	Explanation
D.APP_CODE	The code of the applets and libraries loaded on the card. To be protected from unauthorized modification.
D.APP_C_DATA	Confidential sensitive data of the applications, like the data contained in an object, a static field of a package, a local variable of the currently executed method, or a position of the operand stack. To be protected from unauthorized disclosure.
D.APP_I_DATA	Integrity sensitive data of the applications, like the data contained in an object, a static field of a package, a local variable of the currently executed method, or a position of the operand stack. To be protected from unauthorized modification.
D.APP_KEYS	Cryptographic keys owned by the applets. To be protected from unauthorized disclosure and modification.
D.PIN	Any end-user's PIN. To be protected from unauthorized disclosure and modification.

4.1.2 TSF data

Table 14 TSF data defined in [JC PP]

Asset	Explanation
D.API_DATA	Private data of the API, like the contents of its private fields. To be protected from unauthorized disclosure and modification.
D.CRYPTO	Cryptographic data used in runtime cryptographic computations, like a seed used to generate a key. To be protected from unauthorized disclosure and modification.
D.JCS_CODE	The code of the Java Card System.

Asset	Explanation
	To be protected from unauthorized disclosure and modification.
D.JCS_DATA	The internal runtime data areas necessary for the execution of the Java Card VM, such as, for instance, the frame stack, the program counter, the class of an object, the length allocated for an array, any pointer used to chain data-structures. To be protected from unauthorized disclosure or modification.
D.SEC_DATA	The runtime security data of the Java Card RE, like, for instance, the AIDs used to identify the installed applets, the currently selected applet, the current context of execution and the owner of each object. To be protected from unauthorized disclosure and modification.

Table 15 TSF data defined in this ST

Asset	Explanation
D.CM_DATA	The data of the card management environment, like for instance, the identifiers, the privileges, life cycle states, the memory resource quotas of applets and security domains. To be protected from unauthorized modification.

4.2 Threats

This section introduces the threats to the assets against which specific protection within the TOE or its environment is required. Several groups of threats are distinguished according to the configuration chosen for the TOE and the means used in the attack. The classification is also inspired by the components of the TOE that are supposed to counter each threat.

4.2.1 Confidentiality

Table 16 Confidentiality related threats defined in [JC PP]

Threat	Explanation
T.CONFID-APPLI-DATA	The attacker executes an application to disclose data belonging to another application. See #.CONFID-APPLI-DATA for details. Directly threatened asset(s): D.APP_C_DATA, D.PIN and D.APP_KEYs.
T.CONFID-JCS-CODE	The attacker executes an application to disclose the Java Card System code. See #.CONFID-JCS-CODE for details. Directly threatened asset(s): D.JCS_CODE.
T.CONFID-JCS-DATA	The attacker executes an application to disclose data belonging to the Java Card System. See #.CONFID-JCS-DATA for details. Directly threatened asset(s): D.API_DATA, D.SEC_DATA, D.JCS_DATA and D.CRYPTO.

4.2.2 Integrity

Table 17 Integrity related threats defined in [JC PP]

Threat	Explanation
T.INTEG-APPLI-CODE	The attacker executes an application to alter (part of) its own code or another application's code. See #.INTEG-APPLI-CODE for details. Directly threatened asset(s): D.APP_CODE.
T.INTEG-APPLI-CODE.LOAD	The attacker modifies (part of) its own or another application code when an application package is transmitted to the card for installation. See #.INTEG-APPLI-CODE for details. Directly threatened asset(s): D.APP_CODE.
T.INTEG-APPLI-DATA	The attacker executes an application to alter (part of) another application's data. See #.INTEG-APPLI-DATA for details. Directly threatened asset(s): D.APP_I_DATA, D.PIN and D.APP_KEYs.
T.INTEG-APPLI-DATA.LOAD	The attacker modifies (part of) the initialization data contained in an application package when the package is transmitted to the card for installation. See #.INTEG-APPLI-DATA for details. Directly threatened asset(s): D.APP_I_DATA and D_APP_KEY.
T.INTEG-JCS-CODE	The attacker executes an application to alter (part of) the Java Card System code. See #.INTEG-JCS-CODE for details. Directly threatened asset(s): D.JCS_CODE.
T.INTEG-JCS-DATA	The attacker executes an application to alter (part of) Java Card System or API data. See #.INTEG-JCS-DATA for details. Directly threatened asset(s): D.API_DATA, D.SEC_DATA, D.JCS_DATA and D.CRYPTO. Other attacks are in general related to one of the above, and aimed at disclosing or modifying on-card information. Nevertheless, they vary greatly on the employed means and threatened assets, and are thus covered by quite different objectives in the sequel. That is why a more detailed list is given hereafter.

4.2.3 Identity usurpation

Table 18 Threats defined in [JC PP]

Threat	Explanation
T.SID.1	An applet impersonates another application, or even the Java Card RE, in order to gain illegal access to some resources of the card or with respect to the end user or the terminal. See #.SID for details. Directly threatened asset(s): D.SEC_DATA (other assets may be jeopardized should this attack succeed, for instance, if the identity of the JCRE is usurped), D.PIN and D.APP_KEYs.
T.SID.2	The attacker modifies the TOE's attribution of a privileged role (e.g. default applet and currently selected applet), which allows illegal impersonation of this role. See #.SID for further details.

Threat	Explanation
	Directly threatened asset(s): D.SEC_DATA (any other asset may be jeopardized should this attack succeed, depending on whose identity was forged).

4.2.4 Unauthorized execution

Table 19 Threats defined in [JC PP]

Threat	Explanation
T.EXE-CODE.1	An applet performs an unauthorized execution of a method. See #.EXE-JCS-CODE and #.EXE-APPLI-CODE for details. Directly threatened asset(s): D.APP_CODE.
T.EXE-CODE.2	An applet performs an execution of a method fragment or arbitrary data. See #.EXE-JCS-CODE and #.EXE-APPLI-CODE for details. Directly threatened asset(s): D.APP_CODE.
T.NATIVE	An applet executes a native method to bypass a TOE Security Function such as the firewall. See #.NATIVE for details. Directly threatened asset(s): D.JCS_DATA.

4.2.5 Denial of service

Table 20 Threats defined in [JC PP]

Threat	Explanation
T.RESOURCES	An attacker prevents correct operation of the Java Card System through consumption of some resources of the card: RAM or NVRAM. See #.RESOURCES for details. Directly threatened asset(s): D.JCS_DATA.

4.2.6 Card management

Table 21 Threats defined in [JC PP]

Threat	Explanation
T.DELETION	The attacker deletes an applet or a package already in use on the card, or uses the deletion functions to pave the way for further attacks (putting the TOE in an insecure state). See #.DELETION for details). Directly threatened asset(s): D.SEC_DATA and D.APP_CODE.
T.INSTALL	The attacker fraudulently installs post-issuance of an applet on the card. This concerns either the installation of an unverified applet or an attempt to induce a malfunction in the TOE through the installation process. See #.INSTALL for details. Directly threatened asset(s): D.SEC_DATA (any other asset may be jeopardized should this attack succeed, depending on the virulence of the installed application).

Table 22 Threats defined in this ST

Threat	Explanation
T.COMMUNICATION	The attacker exploits the communication channel established between the TOE and CAD to modify or disclose confidential data. Directly threatened asset(s): D.APP_C_DATA, D.APP_I_DATA, D.APP_KEYS, D.PIN and D.CM_DATA
T.UNAUTHORIZED_CARD_MNGT	The attacker performs unauthorized card management operations (for instance impersonates one of the actor represented on the card) in order to take benefit of the privileges or services granted to this actor on the card such as fraudulent: • load of a package file • installation of a package file • extradition of a package file or an applet • personalization of an applet or a Security Domain • deletion of a package file or an applet • privileges update of an applet or a Security Domain Directly threatened asset(s): D.APP_KEYS, D.APP_C_DATA, D.APP_I_DATA, D.APP_CODE and D.CM_DATA.
T.LIFE_CYCLE	An attacker accesses to an application outside of its expected availability range thus violating irreversible life cycle phases of the application (for instance, an attacker repersonalizes the application). Directly threatened asset(s): D.APP_I_DATA, D.APP_C_DATA and D.CM_DATA.

4.2.7 Services

Table 23 Threats defined in [JC PP]

T.OBJ-DELETION	The attacker keeps a reference to a garbage collected object in order to force the TOE to execute an unavailable method, to make it to crash, or to gain access to a memory containing data that is now being used by another application. See #.OBJ-DELETION for further details. Directly threatened asset(s): D.APP_C_DATA, D.APP_I_DATA and D.APP_KEYs.

4.2.8 Miscellaneous

Table 24 Threats defined in [JC PP]

Threat	Explanation
T.PHYSICAL	The attacker discloses or modifies the design of the TOE, its sensitive data or application code by physical (opposed to logical) tampering means. This threat includes IC failure analysis, electrical probing, unexpected tearing, and DPA. That also includes the modification of the runtime execution of Java Card System or SCP software through alteration of the intended execution order of (set of) instructions through physical tampering techniques. This threatens all the identified assets. This threat refers to the point (7) of the security aspect #.SCP, and all aspects related to confidentiality and integrity of code and data.

Table 25 Threats defined in this ST

Threat	Explanation
T.RNG	An attacker may predict or obtain information about random numbers generated by the TOE.

4.3 Organizational security policies (OSPs)

This section describes the organizational security policies to be enforced with respect to the TOE environment.

Table 26 OSPs defined in [JC PP]

OSP	Explanation
OSP.VERIFICATION	This policy shall ensure the consistency between the export files used in the verification and those used for installing the verified file. The policy must also ensure that no modification of the file is performed in between its verification and the signing by the verification authority. See #.VERIFICATION for details. If the application development guidance provided by the platform developer contains recommandations related to the isolation property of the platform, this policy shall also ensure that the verification authority checks that these recommandations are applied in the application code.

4.4 Assumptions

This section introduces the assumptions made on the environment of the TOE.

Table 27 Assumptions defined in [JC PP]

Assumption	Explanation
A.APPLET	Applets loaded post-issuance do not contain native methods. The Java Card specification explicitly "does not include support for native methods" ([JCVM3], §3.3) outside the API.
A.VERIFICATION	All the bytecodes are verified at least once, before the loading, before the installation or before the execution, depending on the card capabilities, in order to ensure that each bytecode is valid at execution time.

4.5 Statement of compatibility

The statement of compatibility is decribed in the document [SOC]

5 Security objectives

5.1 Security objectives for the TOE

Table 28 Objectives defined in this [JC PP]

Objective	Description
O.SID	The TOE shall uniquely identify every subject (applet, or package) before granting it access to any service.
O.FIREWALL	The TOE shall ensure controlled sharing of data containers owned by applets of different packages or the JCRE and between applets and the TSFs. See #.FIREWALL for details.
O.GLOBAL_ARRAYS_CONFID	The TOE shall ensure that the APDU buffer that is shared by all applications is always cleaned upon applet selection. The TOE shall ensure that the global byte array used for the invocation of the install method of the selected applet is always cleaned after the return from the install method.
O.GLOBAL_ARRAYS_INTEG	The TOE shall ensure that only the currently selected applications may have a write access to the APDU buffer and the global byte array used for the invocation of the install method of the selected applet.
O.NATIVE	The only means that the Java Card VM shall provide for an application to execute native code is the invocation of a method of the Java Card API, or any additional API. See #.NATIVE for details.
O.OPERATE	The TOE must ensure continued correct operation of its security functions. See #.OPERATE for details.
O.REALLOCATION	The TOE shall ensure that the re-allocation of a memory block for the runtime areas of the Java Card VM does not disclose any information that was previously stored in that block.
O.RESOURCES	The TOE shall control the availability of resources for the applications. See #.RESOURCES for details.
O.ALARM	The TOE shall provide appropriate feedback information upon detection of a potential security violation. See #.ALARM for details.
O.CIPHER	The TOE shall provide a means to cipher sensitive data for applications in a secure way. In particular, the TOE must support cryptographic algorithms consistent with cryptographic usage policies and standards. See #.CIPHER for details.
O.KEY-MNGT	The TOE shall provide a means to securely manage cryptographic keys. This concerns the correct generation, distribution, access and destruction of cryptographic keys. See #.KEY-MNGT.
O.PIN-MNGT	The TOE shall provide a means to securely manage PIN objects. See #.PIN-MNGT for details. **NOTE:** PIN objects may play key roles in the security architecture of client applications. The way they are stored and managed in the memory of the smart card must be carefully considered, and this applies to the whole object rather than the sole value of the PIN. For instance, the try counter's value is as sensitive as that of the PIN.

Objective	Description
O.TRANSACTION	The TOE must provide a means to execute a set of operations atomically. See #.TRANSACTION for details.
O.OBJ-DELETION	The TOE shall ensure the object deletion shall not break references to objects. See #.OBJ-DELETION for further details.
O.DELETION	The TOE shall ensure that both applet and package deletion perform as expected. See #.DELETION for details.
O.LOAD	The TOE shall ensure that the loading of a package into the card is safe. Besides, for code loaded post-issuance, the TOE shall verify the integrity and authenticity evidences generated during the verification of the application package by the verification authority. This verification by the TOE shall occur during the loading or later during the install process. **NOTE:** Usurpation of identity resulting from a malicious installation of an applet on the card may also be the result of perturbing the communication channel linking the CAD and the card. Even if the CAD is placed in a secure environment, the attacker may try to capture, duplicate, permute or modify the packages sent to the card. He may also try to send one of its own applications as if it came from the card issuer. Thus, this objective is intended to ensure the integrity and authenticity of loaded CAP files.
O.INSTALL	The TOE shall ensure that the installation of an applet performs as expected (See #.INSTALL for details). Besides, for code loaded post-issuance, the TOE shall verify the integrity and authenticity evidences generated during the verification of the application package by the verification authority. If not performed during the loading process, this verification by the TOE shall occur during the install process.

Note: O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION and O.CIPHER are actually provided to applets in the form of Java Card APIs. Additionally, this Java Card contains vendor-specific libraries for ICAO secure messaging and a special hardened PACE API. Those proprietary libraries are evaluated as part of the TOE.

Table 29 Ojectives defined in this ST

Objective	Explanation
O.CARD-MANAGEMENT	The TOE shall provide card management functionalities (loading, installation, extradition, deletion of applications and GP registry updates) in charge of the life cycle of the whole Java Card and installed applications (applets). The card manager, the application with specific rights responsible for the administration of the smart card, shall control the access to card management functions. It shall also implement the card issuer's policy on card management. **NOTE:** The card manager will be tightly connected in practice with the rest of the TOE, which in return shall very likely rely on the card manager for the effective enforcement of some of its security functions. The mechanism used to ensure authentication of the TOE issuer, that manages the TOE, or of the Service Providers owning a Security Domain with card management privileges is a secure channel. This channel will be used

Objective	Explanation
	afterwards to protect commands exchanged with the TOE in confidentiality and integrity. The platform guarantees that only the ISD or the Service Providers owning a Security Domain with the appropriate privilege (Delegated Management) can manage the applications on the card associated with its Security Domain. This is done accordingly with the card issuer's policy on card management. The actor performing the operation must beforehand authenticate with the Security Domain. In the case of Delegated Management, the card management command will beassociated with an electronic signature (GlobalPlatform token) verified by the ISD before execution.
O.COMMUNICATION	The TOE shall authenticate the origin of the card management requests that the card receives, and authenticate itself to the remote actor. It shall verify the integrity of the card management requests that it receives and be able, when it’s needed, to process card management requests containing encrypted data.The authentication method shall be resistant against replay attacks.
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 (as specified in[ST_IC]), 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

Objective	Explanation
	model is structured and allows for low-level control accesses (segmentation fault detection).
O.SCP.RNG	The TOE will ensure the cryptographic quality of random number generation. For instance random numbers shall not be predictable and shall have sufficient entropy.

5.2 Security objectives for the operational environment

This section introduces the security objectives to be achieved by the environment.

Table 30 Objectives for the operational environment

Objective	Explanaation
OE.APPLET	No applet loaded post-issuance shall contain native methods.
OE.VERIFICATION	All the bytecodes shall be verified at least once, before the loading, before the installation or before the execution, depending on the card capabilities, in order to ensure that each bytecode is valid at execution time. See #.VERIFICATION for details. Additionally, the applet shall follow all the recommendations, if any, mandated in the platform guidance for maintaining the isolation property of the platform. ***NOTE:*** *Constraints to maintain the isolation property of the platform are provided by the platform developer in application development guidance. The constraints apply to all application code loaded in the platform.*
OE.CODE-EVIDENCE	For application code loaded pre-issuance, evaluated technical measures implemented by the TOE or audited organizational measures must ensure that loaded application has not been changed since the code verifications required in OE.VERIFICATION. For application code loaded post-issuance and verified off-card according to the requirements of OE.VERIFICATION, the verification authority shall provide digital evidence to the TOE that the application code has not been modified after the code verification and that he is the actor who performed code verification. For application code loaded post-issuance and partially or entirely verified on-card, technical measures must ensure that the verification required in OE.VERIFICATION is performed. On-card bytecode verifier is out of the scope of this Security Target. **NOTE:** For application code loaded post-issuance and verified off-card, the integrity and authenticity evidence can be achieved by electronic signature of the application code, after code verification, by the actor who performed verification.

5.3 Security objectives rationale

5.3.1 Threats

Table 31 Threats

Threat	Explanation
T.CONFID-APPLI-DATA	This threat is countered by the security objective for the operational environment regarding bytecode verification (OE.VERIFICATION). It is also covered by the isolation commitments stated in the (O.FIREWALL) objective. It relies in its turn on the correct identification of applets stated in (O.SID). Moreover, as the firewall is dynamically enforced, it shall never stop operating, as stated in the (O.OPERATE) objective. As the firewall is a software tool automating critical controls, the objective O.ALARM asks for it to provide clear warning and error messages, so that the appropriate counter-measure can be taken. The objectives O.CARD-MANAGEMENT and O.VERIFICATION contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively. The objectives O.SCP.RECOVERY and O.SCP.SUPPORT are intended to support the O.OPERATE and O.ALARM objectives of the TOE, so they are indirectly related to the threats that these latter objectives contribute to counter. As applets may need to share some data or communicate with the CAD, cryptographic functions are required to actually protect the exchanged information (O.CIPHER). Remark that even if the TOE shall provide access to the appropriate TSFs, it is still the responsibility of the applets to use them. Keys, PIN's are particular cases of an application's sensitive data (the Java Card System may possess keys as well) that ask for appropriate management (O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION). If the PIN class of the Java Card API is used, the objective (O.FIREWALL) shall contribute in covering this threat by controlling the sharing of the global PIN between the applets. Other application data that is sent to the applet as clear text arrives to the APDU buffer, which is a resource shared by all applications. The disclosure of such data is prevented by the security objective O.GLOBAL_ARRAYS_CONFID. Finally, any attempt to read a piece of information that was previously used by an application but has been logically deleted is countered by the O.REALLOCATION objective. That objective states that any information that was formerly stored in a memory block shall be cleared before the block is reused.
T.CONFID-JCS-CODE	This threat is countered by the list of properties described in the (#.VERIFICATION) security aspect. Bytecode verification ensures that each of the instructions used on the Java Card platform is used for its intended purpose and in the intended scope of accessibility. As none of those instructions enables reading a piece of code, no Java Card applet can therefore be executed to disclose a piece of code. Native applications are also harmless because of the objective O.NATIVE, so no application can be run to disclose a piece of code. The (#.VERIFICATION) security aspect is addressed in this ST by the objective for the environment OE.VERIFICATION. The objectives O.CARD-MANAGEMENT and OE.VERIFICATION contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively.

Threat	Explanation
T.CONFID-JCS-DATA	This threat is covered by bytecode verification (OE.VERIFICATION) and the isolation commitments stated in the (O.FIREWALL) security objective. This latter objective also relies in its turn on the correct identification of applets stated in (O.SID). Moreover, as the firewall is dynamically enforced, it shall never stop operating, as stated in the (O.OPERATE) objective. As the firewall is a software tool automating critical controls, the objective O.ALARM asks for it to provide clear warning and error messages, so that the appropriate counter-measure can be taken. The objectives O.CARD-MANAGEMENT and OE.VERIFICATION contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively. The objectives O.SCP.RECOVERY and O.SCP.SUPPORT are intended to support the O.OPERATE and O.ALARM objectives of the TOE, so they are indirectly related to the threats that these latter objectives contribute to counter.
T.INTEG-APPLI-CODE	This threat is countered by the list of properties described in the (#.VERIFICATION) security aspect. Bytecode verification ensures that each of the instructions used on the Java Card platform is used for its intended purpose and in the intended scope of accessibility. As none of these instructions enables modifying a piece of code, no Java Card applet can therefore be executed to modify a piece of code. Native applications are also harmless because of the objective O.NATIVE, so no application can run to modify a piece of code. The (#.VERIFICATION) security aspect is addressed in this configuration by the objective for the environment OE.VERIFICATION. The objectives O.CARD-MANAGEMENT and OE.VERIFICATION contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively. The objective OE.CODE-EVIDENCE contributes to cover this threat by ensuring that integrity and authenticity evidences exist for the application code loaded into the platform.
T.INTEG-APPLI-CODE.LOAD	This threat is countered by the security objective O.LOAD which ensures that the loading of packages is done securely and thus preserves the integrity of packages code. The objective OE.CODE-EVIDENCE contributes to cover this threat by ensuring that the application code loaded into the platform has not been changed after code verification, which ensures code integrity and authenticity. By controlling the access to card management functions such as the installation, update or deletion of applets the objective O.CARD-MANAGEMENT contributes to cover this threat.
T.INTEG-APPLI-DATA	This threat is countered by bytecode verification (OE.VERIFICATION) and the isolation commitments stated in the (O.FIREWALL) objective. This latter objective also relies in its turn on the correct identification of applets stated in (O.SID). Moreover, as the firewall is dynamically enforced, it shall never stop operating, as stated in the (O.OPERATE) objective. As the firewall is a software tool automating critical controls, the objective O.ALARM asks for it to provide clear warning and error messages, so that the appropriate counter-measure can be taken.

Threat	Explanation
	The objectives O.CARD-MANAGEMENT and OE.VERIFICATION contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively. The objective OE.CODE-EVIDENCE contributes to cover this threat by ensuring that the application code loaded into the platform has not been changed after code verification, which ensures code integrity and authenticity. The objectives O.SCP.RECOVERY and O.SCP.SUPPORT are intended to support the O.OPERATE and O.ALARM objectives of the TOE, so they are indirectly related to the threats that these latter objectives contribute to counter. Concerning the confidentiality and integrity of application sensitive data, as applets may need to share some data or communicate with the CAD, cryptographic functions are required to actually protect the exchanged information (O.CIPHER). Remark that even if the TOE shall provide access to the appropriate TSFs, it is still the responsibility of the applets to use them. Keys and PIN's are particular cases of an application's sensitive data (the Java Card System may possess keys as well) that ask for appropriate management (O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION). If the PIN class of the Java Card API is used, the objective (O.FIREWALL) is also concerned. Other application data that is sent to the applet as clear text arrives to the APDU buffer, which is a resource shared by all applications. The integrity of the information stored in that buffer is ensured by the objective O.GLOBAL_ARRAYS_INTEG. Finally, any attempt to read a piece of information that was previously used by an application but has been logically deleted is countered by the O.REALLOCATION objective. That objective states that any information that was formerly stored in a memory block shall be cleared before the block is reused.
T.INTEG-APPLI-DATA.LOAD	This threat is countered by the security objective O.LOAD which ensures that the loading of packages is done securely and thus preserves the integrity of applications data. The objective OE.CODE-EVIDENCE contributes to cover this threat by ensuring that the application code loaded into the platform has not been changed after code verification, which ensures code integrity and authenticity. By controlling the access to card management functions such as the installation, update or deletion of applets the objective O.CARD-MANAGEMENT contributes to cover this threat.
T.INTEG-JCS-CODE	This threat is countered by the list of properties described in the (#.VERIFICATION) security aspect. Bytecode verification ensures that each of the instructions used on the Java Card platform is used for its intended purpose and in the intended scope of accessibility. As none of these instructions enables modifying a piece of code, no Java Card applet can therefore be executed to modify a piece of code. Native applications are also harmless because of the objective O.NATIVE, so no application can be run to modify a piece of code. The (#.VERIFICATION) security aspect is addressed in this configuration by the objective for the environment OE.VERIFICATION. The objectives O.CARD-MANAGEMENT and OE.VERIFICATION contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively.

Threat	Explanation
	The objective OE.CODE-EVIDENCE contributes to cover this threat by ensuring that the application code loaded into the platform has not been changed after code verification, which ensures code integrity and authenticity.
T.INTEG-JCS-DATA	This threat is countered by bytecode verification (OE.VERIFICATION) and the isolation commitments stated in the (O.FIREWALL) objective. This latter objective also relies in its turn on the correct identification of applets stated in (O.SID). Moreover, as the firewall is dynamically enforced, it shall never stop operating, as stated in the (O.OPERATE) objective. As the firewall is a software tool automating critical controls, the objective O.ALARM asks for it to provide clear warning and error messages, so that the appropriate counter-measure can be taken. The objectives O.CARD-MANAGEMENT and OE.VERIFICATION contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively. The objective OE.CODE-EVIDENCE contributes to cover this threat by ensuring that the application code loaded into the platform has not been changed after code verification, which ensures code integrity and authenticity. The objectives O.SCP.RECOVERY and O.SCP.SUPPORT are intended to support the O.OPERATE and O.ALARM objectives of the TOE, so they are indirectly related to the threats that these latter objectives contribute to counter.
T.SID.1	As impersonation is usually the result of successfully disclosing and modifying some assets, this threat is mainly countered by the objectives concerning the isolation of application data (like PINs), ensured by the (O.FIREWALL). Uniqueness of subject-identity (O.SID) also participates to face this threat. It should be noticed that the AIDs, which are used for applet identification, are TSF data. In this configuration, usurpation of identity resulting from a malicious installation of an applet on the card is covered by the objective O.INSTALL. The installation parameters of an applet (like its name) are loaded into a global array that is also shared by all the applications. The disclosure of those parameters (which could be used to impersonate the applet) is countered by the objectives O.GLOBAL_ARRAYS_CONFID and O.GLOBAL_ARRAYS_INTEG. The objective O.CARD-MANAGEMENT contributes, by preventing usurpation of identity resulting from a malicious installation of an applet on the card, to counter this threat.
T.SID.2	This is covered by integrity of TSF data, subject-identification (O.SID), the firewall (O.FIREWALL) and its good working order (O.OPERATE). The objective O.INSTALL contributes to counter this threat by ensuring that installing an applet has no effect on the state of other applets and thus can't change the TOE's attribution of privileged roles. The objectives O.SCP.RECOVERY and O.SCP.SUPPORT are intended to support the O.OPERATE objective of the TOE, so they are indirectly related to the threats that this latter objective contributes to counter.
T.EXE-CODE.1	Unauthorized execution of a method is prevented by the objective OE.VERIFICATION. This threat particularly concerns the point (8) of the security aspect #.VERIFICATION (access modifiers and scope of accessibility for classes, fields and methods). The O.FIREWALL objective is also concerned, because it prevents the execution of non-shareable methods of a class instance by any subject apart from the class instance owner.

Threat	Explanation
T.EXE-CODE.2	Unauthorized execution of a method fragment or arbitrary data is prevented by the objective OE.VERIFICATION. This threat particularly concerns those points of the security aspect related to control flow confinement and the validity of the method references used in the bytecodes.
T.NATIVE	This threat is countered by O.NATIVE which ensures that a Java Card applet can only access native methods indirectly that is, through an API. OE.APPLET also covers this threat by ensuring that no native applets shall be loaded in post-issuance. In addition to this, the bytecode verifier also prevents the program counter of an applet to jump into a piece of native code by confining the control flow to the currently executed method (OE.VERIFICATION).
T.RESOURCES	An attacker prevents correct operation of the Java Card System through consumption of some resources of the card: RAM or NVRAM. See #.RESOURCES for details. Directly threatened asset(s): D.JCS_DATA.
T.DELETION	This threat is covered by the O.DELETION security objective which ensures that both applet and package deletion perform as expected. The objective O.CARD-MANAGEMENT controls the access to card management functions and thus contributes to cover this threat.
T.INSTALL	This threat is covered by the security objective O.INSTALL which ensures that the installation of an applet performs as expected and the security objectives O.LOAD which ensures that the loading of a package into the card is safe. The objective O.CARD-MANAGEMENT controls the access to card management functions and thus contributes to cover this threat.
T.COMMUNICATION	This threat is covered by the O.COMMUNICATION security objective which authenticates the origin of the card management and protects integrity and confidentiality of transmitted data.
T.UNAUTHORIZED_CARD_MNGT	This threat is covered by the following security objectives: • O.CARD-MANAGEMENT controls the access to card management functions such as the loading, installation, extradition or deletion of applets. • O.COMMUNICATION prevents unauthorized users from initiating a malicious card management operation and protects the integrity of the card management data while it is in transit to the Java Card.
T.LIFE_CYCLE	This threat is covered by the security objective O.CARD-MANAGEMENT that controls the access to card management functions such as the loading, installation, extradition or deletion of applets and prevent attacks intended to modify or exploit the current life cycle of applications.
T.OBJ-DELETION	This threat is covered by the O.OBJ-DELETION security objective which ensures that object deletion shall not break references to objects.
T.PHYSICAL	Covered by O.SCP.IC. Physical protections rely on the underlying platform and are therefore an environmental issue.
T.RNG	Covered by O.SCP.RNG

Table 32 Organisational security poliy

OSP.VERIFICATION	This policy is upheld by the security objective of the environment OE.VERIFICATION which guarantees that all the bytecodes shall be verified at least once, before the loading, before the installation or before the execution in order to ensure that each bytecode is valid at execution time. This policy is also upheld by the security objective of the environment OE.CODE-EVIDENCE which ensures that evidences exist that the application code has been verified and not changed after verification.

Table 33 Assumptions

A.APPLET	This assumption is covered by OE.APPLET
A.VERIFICATION	This assumption is covered by OE.VERIFICATION which ensures that bytecode verification is done on the Applet and OE.CODE-EVIDENCE which ensures that the code is not altered after bytecode verification

5.3.2 SPD and security objectives

Table 34 Threats and Security Objectives - Coverage

Threat	Objectives
T.CONFID-APPLI-DATA	As [JC PP] plus remapping from Table 5
T.CONFID-JCS-CODE	As [JC PP] plus remapping from Table 5
T.CONFID-JCS-DATA	As [JC PP] plus remapping from Table 5
T.INTEG-APPLI-CODE	As [JC PP] plus remapping from Table 5
T.INTEG-APPLI-CODE.LOAD	As [JC PP] plus remapping from Table 5
T.INTEG-APPLI-DATA	As [JC PP] plus remapping from Table 5
T.INTEG-APPLI-DATA.LOAD	As [JC PP] plus remapping from Table 5
T.INTEG-JCS-CODE	As [JC PP] plus remapping from Table 5
T.INTEG-JCS-DATA	As [JC PP] plus remapping from Table 5
T.SID.1	As [JC PP] plus remapping from Table 5
T.SID.2	As [JC PP] plus remapping from Table 5
T.EXE-CODE.1	As [JC PP] plus remapping from Table 5
T.EXE-CODE.2	As [JC PP] plus remapping from Table 5
T.NATIVE	As [JC PP] plus remapping from Table 5
T.RESOURCES	As [JC PP] plus remapping from Table 5
T.DELETION	As [JC PP] plus remapping from Table 5
T.INSTALL	As [JC PP] plus remapping from Table 5
T.OBJ-DELETION	As [JC PP] plus remapping from Table 5
T.PHYSICAL	As [JC PP] plus remapping from Table 5
T.COMMUNICATION	O.COMMUNICATION
T.UNAUTHORIZED_CARD_MNGT	O.CARD-MANAGEMENT, O.COMMUNICATION
T.LIFE_CYCLE	O.CARD-MANAGEMENT
T.RNG	O.SCP.RNG

Table 35 Security Objectives and Threats - Coverage

Objective	Threats
O.SID	As [JC PP]
O.FIREWALL	As [JC PP]
O.GLOBAL_ARRAYS_CONFID	As [JC PP]
O.GLOBAL_ARRAYS_INTEG	As [JC PP]
O.NATIVE	As [JC PP]
O.OPERATE	As [JC PP]
O.REALLOCATION	As [JC PP]
O.RESOURCES	As [JC PP]
O.ALARM	As [JC PP]
O.CIPHER	As [JC PP]
O.KEY-MNGT	As [JC PP]
O.PIN-MNGT	As [JC PP]
O.TRANSACTION	As [JC PP]
O.OBJ-DELETION	As [JC PP]
O.DELETION	As [JC PP]
O.LOAD	As [JC PP]
O.INSTALL	As [JC PP]
OE.APPLET	As [JC PP]
OE.VERIFICATION	As [JC PP]
OE.CODE-EVIDENCE	As [JC PP]
O.CARD-MANAGEMENT	T.CONFID-APPLI-DATA, T.CONFID-JCS-CODE, T.CONFID-JCS-DATA, T.INTEG-APPLI-CODE, T.INTEG-APPLI-CODE.LOAD,T.INTEG-APPLI-DATA, T.INTEG-APPLI-DATA.LOAD, T.INTEG-JCS-CODE, T.INTEG-JCS-DATA, T.SID.1, T.DELETION, T.INSTALL
O.SCP.IC	T.PHYSICAL
O.SCP.RECOVERY	T.CONFID-APPLI-DATA, T.CONFID-JCS-DATA, T.INTEG-APPLI-DATA, T.INTEG-JCS-DATA, T.SID.2, T.RESOURCES
O.SCP.SUPPORT	T.CONFID-APPLI-DATA, T.CONFID-JCS-DATA, T.INTEG-APPLI-DATA, T.INTEG-JCS-DATA, T.SID.2, T.RESOURCES
O.SCP.RNG	T.RNG

Table 36 OSPs and Security Objectives - Coverage

OSP	Objectives
OSP.VERIFICATION	As [JC PP]

Table 37 Security Objectives and OSPs - Coverage

Objective	OSPs
O.LOAD	As [JC PP]
OE.VERIFICATION	As [JC PP]

Objective	OSPs
OE.CODE-EVIDENCE	As [JC PP]

Table 38 Assumptions and Security Objectives for the Operational Environment - Coverage

Assumption	Objectives
A.APPLET	As [JC PP]
A.VERIFICATION	As [JC PP]

Table 39 Security Objectives for the Operational Environment and Assumptions - Coverage

Objective	Assumptions
OE.APPLET	As [JC PP]
OE.VERIFICATION	As [JC PP]
OE.CODE-EVIDENCE	As [JC PP]

6 Extended components definition

The following extended SFRs are used in this ST.

- FCS_RNG
  This SFR is defined in [AIS 31], chapter 4.5.1

7 Security requirements

7.1 Notation

The following convention has been used for marking the Common Criteria operations in SFRs.

- Underlined text in SFRs without footnotes denotes CC operations defined in [JC PP]
- Underlined text in SFRs with footnotes denotes CC operations defined in this ST.

Please note that references in square brackets will never be underlined due to technical limitations of the text processing engine.

7.2 Security functional requirements

This section states the security functional requirements for the Java Card System - Open configuration. The groups CoreG_LC, InstG, ADELG, ODELG, CarG have been taken form the Java Card Protection Profile [JC PP].

The group CMGRG has been taken form the Java Card USIM Protection profile [JC USIM PP]. Please note that the SFRs from [JC USIM PP] have been adapted from the USIM profile to the ID profile which is used in this ST. The SFRs from [JC USIM PP] have been reused as inspiration only and neither strict nor demonstrative conformance is claimed to [JC USIM PP].

The group SCPG refers to SFRs which are directly related to SFRs form the Security Traget of the underlying hardware platform [ST_IC].

Table 40 SFR groups

Group	Description
Core with Logical Channels (CoreG_LC)	The CoreG_LC contains the requirements concerning the runtime environment of the Java Card System implementing logical channels. This includes the firewall policy and the requirements related to the Java Card API. Logical channels are a Java Card specification version 3.0.5 feature. This group is the union of requirements from the Core (CoreG) and the Logical channels (LCG) groups defined in [JC PP].
Installation (InstG)	The InstG contains the security requirements concerning the installation of post-issuance applications. It does not address card management issues in the broad sense, but only those security aspects of the installation procedure that are related to applet execution.
Applet deletion (ADELG)	The ADELG contains the security requirements for erasing installed applets from the card, a feature introduced in Java Card specification version 3.0.5.
Object deletion (ODELG)	The ODELG contains the security requirements for the object deletion capability. This provides a safe memory recovering mechanism. This is a Java Card specification version 3.0.5 feature.
Secure carrier (CarG)	The CarG group has been refined and extended by the security requirements of the CMGRG group
Card manager (CMGRG)	The CMGRG group contains the security requirements for the GlobalPlatform card manager.
Hardware IC support (SCPG)	This group contains security requirements which are directly related to the underlying hardware platform.

The SFRs refer to all potentially applicable subjects, objects, information, operations and security attributes.

Subjects are active components of the TOE that (essentially) act on the behalf of users. The users of the TOE include people or institutions (like the applet developer, the card issuer, the verification authority), hardware (like the CAD where the card is inserted or the PCD) and software components (like the application packages installed on the card). Some of the users may just be aliases for other users. For instance, the verification authority in charge of the bytecode verification of the applications may be just an alias for the card issuer.

Note: The content of Table 41 to Table 45 is copied from [JC PP]. It does not contain the respective elements from the card manager functionality added in this ST (i.e. section 7.2.6). The subject S.BCV has been removed because the byte code verifier belongs to the operational environment and is therefor not a valid subject for this TOE.

Table 41 Subjects

Subjects	Description
S.ADEL	The applet deletion manager which also acts on behalf of the card issuer. It may be an applet ([JCRE3], §11), but its role asks anyway for a specific treatment from the security viewpoint. This subject is unique and is involved in the ADEL security policy defined in [JC PP], section 7.1.3.
S.APPLET	Any applet instance.
S.CAD	The CAD represents off-card entity that communicates with the S.INSTALLER.
S.INSTALLER	The installer is the on-card entity which acts on behalf of the card issuer. This subject is involved in the loading of packages and installation of applets.
S.JCRE	The runtime environment under which Java programs in a smart card are executed.
S.JCVM	The bytecode interpreter that enforces the firewall at runtime.
S.LOCAL	Operand stack of a JCVM frame, or local variable of a JCVM frame containing an object or an array of references.
S.MEMBER	Any object's field, static field or array position.
S.PACKAGE	A package is a namespace within the Java programming language that may contain classes and interfaces, and in the context of Java Card technology, it defines either a user library, or one or several applets.

Table 42 Objects

Object	Description
O.APPLET	Any installed applet, its code and data.
O.CODE_PKG	The code of a package, including all linking information. On the Java Card platform, a package is the installation unit.
O.JAVAOBJECT	Java class instance or array. It should be noticed that KEYS, PIN, arrays and applet instances are specific objects in the Java programming language.

Table 43 Information

Information	Description
I.APDU	Any APDU sent to or from the card through the communication channel.
I.DATA	JCVM Reference Data: objectref addresses of APDU buffer, JCRE-owned instances of APDU class and byte array for install method.

Security attributes linked to these subjects, objects and information are described in the following table with their values:

Table 44 Security attributes

Security attribute	Description/Value
Active Applets	The set of the active applets' AIDs. An active applet is an applet that is selected on at least one of the logical channels.
Applet Selection Status	"Selected" or "Deselected".
Applet's version number	The version number of an applet (package) indicated in the export file.
Class	Identifies the implementation class of the remote object.
Context	Package AID or "Java Card RE". Currently Active
Currently Active Context	Package AID or "Java Card RE".
Dependent package AID	Allows the retrieval of the Package AID and Applet's version number ([JCVM3], §4.5.2).
ExportedInfo	Boolean (indicates whether the remote object is exportable or not).
Identifier	The Identifier of a remote object or method is a number that uniquely identifies the remote object or method, respectively.
LC Selection Status	Multiselectable, Non-multiselectable or "None".
LifeTime	CLEAR_ON_DESELECT or PERSISTENT (*).
Owner	The Owner of an object is either the applet instance that created the object or the package (library) where it has been defined (these latter objects can only be arrays that initialize static fields of the package). The owner of a remote object is the applet instance that created the object.
Package AID	The AID of each package indicated in the export file.
Registered Applets	The set of AID of the applet instances registered on the card.
Resident Packages	The set of AIDs of the packages already loaded on the card.
Selected Applet Context	Package AID or "None".
Sharing	Standards, SIO, Java Card RE entry point or global array.
Static References	Static fields of a package may contain references to objects. The Static References attribute records those references.

(*) Transient objects of type CLEAR_ON_RESET behave like persistent objects in that they can be accessed only when the Currently Active Context is the object's context.

Table 45 Operations

Operation	Description
OP.ARRAY_ACCESS(O.JAVAOBJECT, field)	Read/Write an array component.
OP.CREATE(Sharing, LifeTime) (*)	Creation of an object (new or makeTransient call).
OP.DELETE_APPLET (O.APPLET, ...)	Delete an installed applet and its objects, either logically or physically.

Operation	Description
OP.DELETE_PCKG (O.CODE_PKG, ...)	Delete a package, either logically or physically.
OP.INSTANCE_FIELD(O.JAVAOBJECT, field)	Read/Write a field of an instance of a class in the Java programming language.
OP.INVK_VIRTUAL (O.JAVAOBJECT, method, arg1, ...)	Invoke a virtual method (either on a class instance or an array object).
OP.INVK_INTERFACE (O.JAVAOBJECT, method, arg1, ...)	Invoke an interface method.
OP.JAVA (...)	Any access in the sense of [JCRE3], §6.2.8. It stands for one of the operations OP.ARRAY_ACCESS, OP.INSTANCE_FIELD, OP.INVK_VIRTUAL, OP.INVK_INTERFACE, OP.THROW and OP.TYPE_ACCESS.
OP.PUT(S1,S2,I)	Transfer a piece of information I from S1 to S2.
OP.THROW(O.JAVAOBJECT)	Throwing of an object (athrow, see [JCRE3], §6.2.8.7).
OP.TYPE_ACCESS(O.JAVAOBJECT, class)	Invoke checkcast or instanceof on an object in order to access to classes (standard or shareable interfaces objects).

Operations (prefixed with "OP") are described in the following table. Each operation has parameters given between brackets, among which there is the "accessed object", the first one, when applicable. Parameters may be seen as security attributes that are under the control of the subject performing the operation.

(*) For this operation, there is no accessed object. This rule enforces that shareable transient objects are not allowed. For instance, during the creation of an object, the Java Card class attribute's value is chosen by the creator.

7.2.1 CoreG_LC security functional requirements

7.2.1.1 FDP_ACC.2/FIREWALL Complete access control

FDP_ACC.2.1/FIREWALL

The TSF shall enforce the FIREWALL access control SFP on S.PACKAGE, S.JCRE, S.JCVM, O.JAVAOBJECT and all operations among subjects and objects covered by the SFP.

Refinement: The operations involved in the policy are:

- OP.CREATE
- OP.INVK_INTERFACE
- OP.INVK_VIRTUAL
- OP.JAVA
- OP.THROW
- OP.TYPE_ACCESS

FDP_ACC.2.2/FIREWALL

The TSF shall ensure that all operations between any subject controlled by the TSF and any object controlled by the TSF are covered by an access control SFP.

> *Note: It should be noticed that accessing array's components of a static array, and more generally fields and methods of static objects, is an access to the corresponding O.JAVAOBJECT.*

7.2.1.2 FDP_ACF.1/FIREWALL Security attribute based access control

FDP_ACF.1.1/FIREWALL

The TSF shall enforce the FIREWALL access control SFP to objects based on the following:

Table 46 Security attributes

Subjects/objects	Security attributes
S.PACKAGE	LC Selection Status
S.JCVM	Active Applets, Currently Active Context
S.JCRE	Selected Applet Context
O.JAVAOBJECT	Sharing, Context, LifeTime

FDP_ACF.1.2/FIREWALL

The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

- R.JAVA.1 ([JCRE3], §6.2.8): S.PACKAGE may freely perform OP.ARRAY_ACCESS, OP.INSTANCE_FIELD, OP.INVK_VIRTUAL, OP.INVK_INTERFACE, OP.THROW or OP.TYPE_ACCESS upon any O.JAVAOBJECT whose Sharing attribute has value "JCRE entry point" or "global array".
- R.JAVA.2 ([JCRE3], §6.2.8): S.PACKAGE may freely perform OP.ARRAY_ACCESS, OP.INSTANCE_FIELD, OP.INVK_VIRTUAL, OP.INVK_INTERFACE or OP.THROW upon any O.JAVAOBJECT whose Sharing attribute has value "Standard" and whose Lifetime attribute has value "PERSISTENT" only if O.JAVAOBJECT's Context attribute has the same value as the active context.
- R.JAVA.3 ([JCRE3], §6.2.8.10): S.PACKAGE may perform OP.TYPE_ACCESS upon an O.JAVAOBJECT whose Sharing attribute has value "SIO" only if O.JAVAOBJECT is being cast into (checkcast) or is being verified as being an instance of (instanceof) an interface that extends the Shareable interface.
- R.JAVA.4 ([JCRE3], §6.2.8.6): S.PACKAGE may perform OP.INVK_INTERFACE upon an O.JAVAOBJECT whose Sharing attribute has the value "SIO", and whose Context attribute has the value "Package AID", only if the invoked interface method extends the Shareable interface and one of the following conditions applies:
  - The value of the attribute Selection Status of the package whose AID is "Package AID" is "Multiselectable",
  - The value of the attribute Selection Status of the package whose AID is "Package AID" is "Non-multiselectable", and either "Package AID" is the value of the currently selected applet or otherwise "Package AID" does not occur in the attribute Active Applets.
- R.JAVA.5: S.PACKAGE may perform OP.CREATE only if the value of the Sharing parameter is "Standard".

FDP_ACF.1.3/FIREWALL

The TSF shall explicitly authorise access of subjects to objects based on the following additional rules:

- 1) The subject S.JCRE can freely perform OP.JAVA(") and OP.CREATE, with the exception given in FDP_ACF.1.4/FIREWALL, provided it is the Currently Active Context.
- 2) The only means that the subject S.JCVM shall provide for an application to execute native code is the invocation of a Java Card API method (through OP.INVK_INTERFACE or OP.INVK_VIRTUAL).

FDP_ACF.1.4/FIREWALL

The TSF shall explicitly deny access of subjects to objects based on the following additional rules:

1. Any subject with OP.JAVA upon an O.JAVAOBJECT whose LifeTime attribute has value "CLEAR_ON_DESELECT" if O.JAVAOBJECT's Context attribute is not the same as the Selected Applet Context.
2. Any subject attempting to create an object by the means of OP.CREATE and a "CLEAR_ON_DESELECT" LifeTime parameter if the active context is not the same as the Selected Applet Context.

7.2.1.3 FDP_IFC.1/JCVM Subset information flow control

FDP_IFC.1.1/JCVM

The TSF shall enforce the JCVM information flow control SFP on S.JCVM, S.LOCAL, S.MEMBER, I.DATA and OP.PUT(S1, S2, I).

7.2.1.4 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:

Table 47

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:

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

FDP_IFF.1.3/JCVM [Editorially refined]

The TSF shall enforce no additional rule[1].

FDP_IFF.1.4/JCVM

The TSF shall explicitly authorise an information flow based on the following rules: none[2].

FDP_IFF.1.5/JCVM

The TSF shall explicitly deny an information flow based on the following rules: none[3].

[1] [assignment: additional information flow control SFP rules]

[2] [assignment: rules, based on security attributes, that explicitly authorise information flows]

[3] [assignment: rules, based on security attributes, that explicitly deny information flows]

7.2.1.5 FDP_RIP.1/OBJECTS Subset residual information protection

FDP_RIP.1.1/OBJECTS

The TSF shall ensure that any previous information content of a resource is made unavailable upon the allocation of the resource to the following objects: class instances and arrays.

> *Note: The semantics of the Java programming language requires for any object field and array position to be initialized with default values when the resource is allocated [JCVM3], §2.5.1.*

7.2.1.6 FMT_MSA.1/JCRE Management of security attributes

FMT_MSA.1.1/JCRE

The TSF shall enforce the FIREWALL access control SFP to restrict the ability to modify the security attributes Selected Applet Context to the Java Card RE.

> *Note: The modification of the Selected Applet Context should be performed in accordance with the rules given in[JCRE3], §4 and[JCVM3], §3.4.*

7.2.1.7 FMT_MSA.1/JCVM Management of security attributes

FMT_MSA.1.1/JCVM

The TSF shall enforce the FIREWALL access control SFP and the JCVM information flow control SFP to restrict the ability to modify the security attributes Currently Active Context and Active Applets to the Java Card VM (S.JCVM).

> *Note: The modification of the Currently Active Context should be performed in accordance with the rules given in[JCRE3], §4 and[JCVM3], §3.4.*

7.2.1.8 FMT_MSA.2/FIREWALL_JCVM Secure security attributes

FMT_MSA.2.1/FIREWALL_JCVM

The TSF shall ensure that only secure values are accepted for all the security attributes of subjects and objects defined in the FIREWALL access control SFP and the JCVM information flow control SFP.

7.2.1.9 FMT_MSA.3/FIREWALL Static attribute initialisation

FMT_MSA.3.1/FIREWALL

The TSF shall enforce the FIREWALL access control SFP to provide restrictive default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/FIREWALL [Editorially Refined]

The TSF shall not allow any role to specify alternative initial values to override the default values when an object or information is created.

7.2.1.10 FMT_MSA.3/JCVM Static attribute initialization

FMT_MSA.3.1/JCVM

The TSF shall enforce the JCVM information flow control SFP to provide restrictive default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/JCVM [Editorially Refined]

The TSF shall not allow any role to specify alternative initial values to override the default values when an object or information is created.

7.2.1.11 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1

The TSF shall be capable of performing the following management functions:

- Modify the Currently Active Context, the Selected Applet Context and the Active Applets.

7.2.1.12 FMT_SMR.1 Security roles

FMT_SMR.1.1

The TSF shall maintain the roles

- Java Card RE (JCRE),
- Java Card VM (JCVM).

FMT_SMR.1.2

The TSF shall be able to associate users with roles.

7.2.1.13 FCS_CKM.1 Cryptographic key generation

FCS_CKM.1.1/RSA

The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm for RSA (with or w/o CRT)[1] and specified cryptographic key sizes 512-2048 bit[2] that meet the following:

- According to section 3.2(1) in [PKCS v2.2]
- According to section 3.2(2) in [PKCS v2.2], for u=2, i.e. with 2 primes only[3]

FCS_CKM.1.1/EC

The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm for EC[4] and specified cryptographic key sizes 224, 256, 384, 512, 521[5] bit that meet the following:

- According to chapters 4.3.3 and 4.3.3.2 the appendix A4.3 in [X9.62].
- According to section 6.4.2 “Generation of signature key and verification key” in [ISO14888-3].
- According to appendix A.16.9 “An algorithm for generating EC keys” in [IEEE P1363][6]

[1] [assignment: cryptographic key generation algorithm

[2] [assignment: cryptographic key sizes]

[3] [assignment: list of standards]

[4] [assignment: cryptographic key generation algorithm]

[5] [assignment: cryptographic key sizes]

[6] [assignment: list of standards]

Note: The certification covers the following curves:
NIST standard curves from [FIPS 186-3]:
P224, P256, P384, P521
Brainpool curves from [RFC 5639]:
BrainpoolP224r1, BrainpoolP256r1, BrainpoolP320r1, BrainpoolP384r1, BrainpoolP512r1, BrainpoolP224t1, BrainpoolP256t1, BrainpoolP320t1,BrainpoolP384t1, BrainpoolP512t1

7.2.1.14 FCS_CKM.2 Cryptographic key distribution

FCS_CKM.2.1

The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method set*** methods for TDES, AES, EC, RSA and copyDomainParametersFrom method for EC in javacard.security package[1] that meets the following [JCAPI3][2].

7.2.1.15 FCS_CKM.3 Cryptographic key access

FCS_CKM.3.1

The TSF shall perform access of TDES, AES, EC, RSA keys[3] in accordance with a specified cryptographic key access method get*** methods in package javacard.security[4] that meets the following [JCAPI3][5].

7.2.1.16 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 clearKey method[6] that meets the following [JCAPI3][7].

7.2.1.17 FCS_COP.1 Cryptographic operation

Note: The document [SECGUIDE] provides recommendations for the secure usage of the algorithms listed in this paragraph.

FCS_COP.1.1/JCAPI

The TSF shall perform cryptographic operations from Table 48[8] in accordance with a specified cryptographic algorithm from Table 48[9] and cryptographic key sizes from Table 48[10] that meet the following standards from Table 48[11].

[1] [assignment: cryptographic key distribution method]
[2] [assignment: list of standards]
[3] [assignment: type of cryptographic key access]
[4] [assignment: cryptographic key access method]
[5] [assignment: list of standards]
[6] [assignment: cryptographic key destruction method]
[7] [assignment: list of standards]
[8] [assignment: list of cryptographic operations]
[9] [assignment: cryptographic algorithm]
[10] [assignment: cryptographic key sizes]
[11] [assignment: list of standards]

Table 48 JCAPI algorithms

CC iteration	cryptographic operations	cryptographic algorithm	cryptographic key sizes	standards
/TDES-ENC	TDES encryption and decryption	ECB mode w/o padding and with ISO/IEC 9797-1 padding method 1 and 2	112, 168	[SP800-67] [SP800-38A] [ISO9797-1]
		CBC mode w/o padding and with ISO/IEC 9797-1 padding method 1 and 2	112, 168	[SP800-67] [SP800-38A] [ISO9797-1]
/AES-ENC	AES encryption and decryption	ECB mode w/o padding and with ISO/IEC 9797-1 padding method 1 and 2	128, 192, 256	[FIPS 197] [SP800-38A] [ISO9797-1]
		CBC mode w/o padding and with ISO/IEC 9797-1 padding method 1 and 2	128, 192, 256	[FIPS 197] [SP800-38A] [ISO9797-1]
/TDES-MAC	TDES MAC calculation and verification	ISO/IEC 9797-1 algorithm 3 and padding method 2 with 8 bytes MAC	112	[SP800-67] [ISO9797-1]
		ISO/IEC 9797-1 algorithm 1 and padding method 1 and 2 with 8 bytes MAC	112, 168	[SP800-67] [ISO9797-1]
/AES-MAC	AES MAC calculation and verification	ISO/IEC 9797-1 algorithm 1 without padding	128, 192, 256	[FIPS 197] [ISO9797-1]
		CMAC	128, 192, 256	[FIPS 197] [SP800-38B]
/HASH	hash calculation	SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	None	[FIPS 180-4]
/RSA-DEC	RSA decryption	RSADP	see note	[PKCS v2.2]
/RSA-ENC	RSA encryption	RSAEP	see note	[PKCS v2.2]

CC iteration	cryptographic operations	cryptographic algorithm	cryptographic key sizes	standards
/RSA-PKCS1-DEC	RSA decryption	RSAES-PKCS1-V1_5	see note	[PKCS v1.5]
/RSA-PKCS1-ENC	RSA encryption	RSAES-PKCS1-V1_5	see note	[PKCS v1.5]
/RSA-SIG	RSA signature generation	RSASSA-PSS w/o hash and with hash in {SHA-1, SHA224, SHA256, SHA384, SHA512}}	see note	[PKCS v2.2]
		RSASSA-PKCS-v1_5 w/o hash and with hash in {SHA-1, SHA224, SHA256, SHA384, SHA512}	see note	[PKCS v2.2]
		ISO 9796-2, scheme 1 with SHA-1	see note	[ISO9796-2]
/RSA-VER	RSA signature verification	RSASSA-PSS w/o hash and with hash in {SHA-1, SHA224, SHA256, SHA384, SHA512}	see note	[PKCS v2.2]
		RSASSA-PKCS-v1_5 w/o hash and with hash in {SHA-1, SHA224, SHA256, SHA384, SHA512}	see note	[PKCS v2.2]
		ISO 9796-2, scheme 1 with SHA-1	see note	[ISO9796-2]
/ECDSA-SIG	EC signature generation	ECDSA sign w/o hash and with hash in {SHA-1, SHA224, SHA256, SHA384, SHA512}	see note	[SEC1]
/ECDSA-VER	EC signature verification	ECDSA verify w/o hash and with hash in {SHA-1, SHA224, SHA256, SHA384, SHA512}	see note	[SEC1]
/ECDH	EC Key agreement	ECSVDP-DH	see note	[IEEE P1363]
		ALG_EC_SVDP_DH_PLAIN	see note	[JCAPI3]
		ALG_EC_SVDP_DH_PLAIN_XY	see note	[JCAPI3]
/PACE	password authenticated key agreement	ALG_EC_PACE_GM	see note	[JCAPI3]

Note: Cofactor multiplication is not supported for ECDH. All certified curves in this ST have cofactor = 1 and therefore cofactor multiplication is not necessary.

Note: The certified curves and bit length for ECDSA-SIG, ECDSA-VER, ECDH and PACE are listed in FCS_CKM.1.1/EC in chapter 7.2.1.13.

Note: SHA 384 and SHA 512 are supported but have no security hardening and shall be used only for hashing of non security critical data.

Note: Key sizes for RSA-DEC, RSA-ENC, RSA-PKCS1-DEC, RSA-PKCS1-ENC, RSA-SIG and RSA-VER are dependent on the configured RSA library. For RSA 2K the certified key size range is 512-2048 bits and for RSA 4K the certified key size range is 512-4096 bits.

Note: The RSA 2K and RSA 4K options support RSA-DEC, RSA-PKCS1-DEC, RSA-SIG key sizes 512- 2048 bits with exponent/modulus and CRT calculation. The RSA 4K option supports RSA-DEC, RSA-PKCS1-DEC and RSA-SIG with keysizes from 2049-4096 bits with CRT calculation only.

FCS_COP.1.1/SCP

The TSF shall perform cryptographic operations from Table 49[1] in accordance with a specified cryptographic algorithm from Table 49[2] and cryptographic key sizes from Table 49[3] that meet the following standards from Table 49[4].

Table 49 GP cryptographic algorithms

CC iteration	cryptographic operations	cryptographic algorithm	cryptographic key sizes	Standards
/ENC-TDES	secure channel decryption	SCP02, option 15 and 55	112	[GP v23]
/ENC-AES	secure channel decryption	SCP03, option 10 and 00	128,192,256	[GPv23 Amd D]
/MAC-TDES	secure channel authentication	SCP02, option 15 and 55	112	[GP v23]
/MAC-AES	secure channel authentication	SCP03, option 10 and 00	128,192,256	[GPv23 Amd D]

Note: ENC-TDES and ENC-AES supports command decryption/unwrap only

FCS_COP.1.1/SM

The TSF shall perform cryptographic operations from Table 50[5] in accordance with a specified cryptographic algorithm from Table 50[6] and cryptographic key sizes from Table 50[7] that meet the following standards from Table 50[8]

Table 50 ICAO cryptographic algorithms

CC iteration	cryptographic operations	cryptographic algorithm	cryptographic key sizes	Standards
/ICAO-ENC-TDES	secure messaging encryption	TDES in CBC mode	112	[DOC9303]
/ICAO-MAC-TDES	secure messaging authentication	Retail MAC with TDES	112	[DOC9303]
/ICAO-ENC-AES	secure messaging encryption	AES in CBC mode	128,192,256	[DOC9303]

[1] [assignment: cryptographic operations]

[2] [assignment: cryptographic algorithms]

[3] [assignment: cryptographic key sizes]

[4] [assignment: list of standards]

[5] [assignment: list of cryptographic operations]

[6] [assignment: cryptographic algorithm]

[7] [assignment: cryptographic key sizes]

[8] [assignment: list of standards]

CC iteration	cryptographic operations	cryptographic algorithm	cryptographic key sizes	Standards
/ICAO-MAC-AES	secure messaging authentication	C-MAC with 8 bytes	128,192,256	[DOC9303]
/18013-ENC-AES	secure messaging encryption	AES in CBC mode	128,192,256	[ISO18013-3]
/18013-MAC-AES	secure messaging authentication	C-MAC with 8 bytes	128,192,256	[ISO18013-3]
/PACE	password authenticated key agreement	PACE with generic mapping	see note	[DOC9303] chapter 4.1

Note: The PACE implementation according to this SFR has special countermeasures against side channel leakage such that it can be used for low entropy secret passwords.

7.2.1.18 FDP_RIP.1/ABORT Subset residual information protection

FDP_RIP.1.1/ABORT

The TSF shall ensure that any previous information content of a resource is made unavailable upon the deallocation of the resource from the following objects: any reference to an object instance created during an aborted transaction.

7.2.1.19 FDP_RIP.1/APDU Subset residual information protection

FDP_RIP.1.1/APDU

The TSF shall ensure that any previous information content of a resource is made unavailable upon the allocation of the resource to the following objects: the APDU buffer.

7.2.1.20 FDP_RIP.1/bArray Subset residual information protection

FDP_RIP.1.1/bArray

The TSF shall ensure that any previous information content of a resource is made unavailable upon the deallocation of the resource from the following objects: the bArray object.

7.2.1.21 FDP_RIP.1/KEYS Subset residual information protection

FDP_RIP.1.1/KEYS

The TSF shall ensure that any previous information content of a resource is made unavailable upon the deallocation of the resource from the following objects: the cryptographic buffer (D.CRYPTO).

7.2.1.22 FDP_RIP.1/TRANSIENT Subset residual information protection

FDP_RIP.1.1/TRANSIENT

The TSF shall ensure that any previous information content of a resource is made unavailable upon the deallocation of the resource from the following objects: any transient object.

7.2.1.23 FDP_ROL.1/FIREWALL Basic rollback

FDP_ROL.1.1/FIREWALL

The TSF shall enforce the FIREWALL access control SFP and the JCVM information flow control SFP to permit the rollback of the operations OP.JAVA and OP.CREATE on the object O.JAVAOBJECT.

FDP_ROL.1.2/FIREWALL

The TSF shall permit operations to be rolled back within the scope of a select(), deselect(), process(), install() or uninstall() call, notwithstanding the restrictions given in [JCRE3] §7.7, within the bounds of the Commit Capacity ([JCRE3] §7.8), and those described in [JCRE3].

7.2.1.24 FAU_ARP.1 Security alarms

FAU_ARP.1.1

The TSF shall take one of the following actions:

- throw an exception,
- lock the card session,
- reinitialize the Java Card System and its data,
- none[1]

upon detection of a potential security violation.

Refinement:

The "potential security violation" stands for one of the following events:

- CAP file inconsistency,
- typing error in the operands of a bytecode,
- applet life cycle inconsistency,
- card tearing (unexpected removal of the Card out of the CAD) and power failure,
- abort of a transaction in an unexpected context, (see abortTransaction() [JCAPI3] and [JCRE3] §7.6.2)
- violation of the Firewall or JCVM SFPs,
- unavailability of resources,
- array overflow,
- list of other runtime errors[2].
  - stack overflow
  - illegal methods arguments
  - integrity check failure

7.2.1.25 FDP_SDI.2 Stored data integrity monitoring and action

FDP_SDI.2.1

[1] [assignment: list of other actions]

[2] [assignment: list of other runtime errors]

The TSF shall monitor user data stored in containers controlled by the TSF for integrity errors[1] on all objects, based on the following attributes: Checksum associated to keys and pins[2].

FDP_SDI.2.2

Upon detection of a data integrity error, the TSF shall perform a security reset[3].

7.2.1.26 FPR_UNO.1 Unobservability

FPR_UNO.1.1

The TSF shall ensure that all users and subjects[4] are unable to observe the operation OP.JAVA(…) for cryptographic API methods[5] on D.APP_KEYS, D.JCS_KEYS, D.PIN, D.CRYPTO[6] by any subject[7].

7.2.1.27 FPT_FLS.1 Failure with preservation of secure state

FPT_FLS.1.1

The TSF shall preserve a secure state when the following types of failures occur: those associated to the potential security violations described in 7.2.1.24.

7.2.1.28 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

- the rules defined in [JCVM3] specification,
- the API tokens defined in the export files of reference implementation,
- none[8]

when interpreting the TSF data from another trusted IT product.

7.2.1.29 FIA_ATD.1/AID User attribute definition

FIA_ATD.1.1/AID

The TSF shall maintain the following list of security attributes belonging to individual users:

- Package AID,

[1] [assignment:integrity errors]

[2] [assignment: user data attributes]

[3] [assignment: action to be taken]

[4] [assignment: list of users and/or subjects]

[5] [assignment: list of operations]

[6] [assignment: list of objects]

[7] [assignment: list of protected users and/or subjects]

[8] [assignment: list of interpretation rules to be applied by the TSF]

- Applet's version number,
- Registered applet AID,
- Applet Selection Status ([JCVM3] §6.5).

Note: Refinement: "Individual users" stand for applets.

7.2.1.30 FIA_UID.2/AID User identification before any action

FIA_UID.2.1/AID

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

7.2.1.31 FIA_USB.1/AID User-subject binding

FIA_USB.1.1/AID

The TSF shall associate the following user security attributes with subjects acting on the behalf of that user: Package AID.

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: The rules defined in [JCRE3], chapter 11[1].

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: The rules defined in [JCRE3], chapter 11[2].

7.2.1.32 FMT_MTD.1/JCRE Management of TSF data

FMT_MTD.1.1/JCRE

The TSF shall restrict the ability to modify the list of registered applets' AIDs to the JCRE.

7.2.1.33 FMT_MTD.3/JCRE Secure TSF data

FMT_MTD.3.1/JCRE

The TSF shall ensure that only secure values are accepted for the registered applets' AIDs.

7.2.2 InstG security functional requirements

This group consists of the SFRs related to the installation of the applets, which addresses security aspects outside the runtime. The installation of applets is a critical phase, which lies partially out of the boundaries of the firewall, and therefore requires specific treatment. In this ST, loading a package or installing an applet is

[1] [assignment: rules for the initial association of attributes]

[2] [assignment: rules for the changing of attributes]

modeled as importation of user data (that is, user application's data) with its security attributes (such as the parameters of the applet used in the firewall rules).

7.2.2.1 FDP_ITC.2/Installer Import of user data with security attributes

FDP_ITC.2.1/Installer [Editorially Refined]

The TSF shall enforce the Secure Channel Protocol information flow control policy when importing user data, controlled under the SFP, from outside of the TOE.

Note: The original SFR defined in [JC PP] relates to the "PACKAGE LOADING information flow control SFP", which is defined in FDP_IFC.1/CM of [JC PP]. This Security Target has refined the SFR FDP_IFC.1/CM to by the SFR FDP_IFC.2/SC. The security policy in FDP_IFC.2/SC is called "Secure Channel Protocol information flow control policy" (see chapter 7.2.6.14).

FDP_ITC.2.2/Installer

The TSF shall use the security attributes associated with the imported user data.

FDP_ITC.2.3/Installer

The TSF shall ensure that the protocol used provides for the unambiguous association between the security attributes and the user data received.

FDP_ITC.2.4/Installer

The TSF shall ensure that interpretation of the security attributes of the imported user data is as intended by the source of the user data.

FDP_ITC.2.5/Installer

The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE:

Package loading is allowed only if, for each dependent package, its AID attribute is equal to a resident package AID attribute, the major (minor) Version attribute associated to the dependent package is lesser than or equal to the major (minor) Version attribute associated to the resident package ([JCVM3], §4.5.2).

7.2.2.2 FMT_SMR.1/Installer Security roles

FMT_SMR.1.1/Installer

The TSF shall maintain the roles**:** Installer.

FMT_SMR.1.2/Installer

The TSF shall be able to associate users with roles.

7.2.2.3 FPT_RCV.3/Installer Automated recovery without undue loss

FPT_RCV.3.1/Installer

When automated recovery from none[1] is not possible, the TSF shall enter a maintenance mode where the ability to return to a secure state is provided.

[1] [assignment: list of failures/service discontinuities]

FPT_RCV.3.2/Installer

For any failures/service discontinuities[1], 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 null[2] 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. FMT_SMR.1/Installer

7.2.2.4 FPT_FLS.1/Installer Failure with preservation of secure state

FPT_FLS.1.1/Installer

The TSF shall preserve a secure state when the following types of failures occur: the installer fails to load/install a package/applet as described in[JCRE3] §11.1.5.

7.2.3 AdelG security functional requirements

This group consists of the SFRs related to the deletion of applets and/or packages, enforcing the applet deletion manager (ADEL) policy on security aspects outside the runtime. Deletion is a critical operation and therefore requires specific treatment. This policy is better thought as a frame to be filled by ST implementers.

7.2.3.1 FDP_ACC.2/ADEL Complete access control

FDP_ACC.2.1/ADEL

The TSF shall enforce the ADEL access control SFP on S.ADEL, S.JCRE, S.JCVM, O.JAVAOBJECT, O.APPLET and O.CODE_PKG and all operations among subjects and objects covered by the SFP.

Refinement:

The operations involved in the policy are:

- OP.DELETE_APPLET,
- OP.DELETE_PCKG,
- OP.DELETE_PCKG_APPLET.

FDP_ACC.2.2/ADEL

The TSF shall ensure that all operations between any subject controlled by the TSF and any object controlled by the TSF are covered by an access control SFP.

7.2.3.2 FDP_ACF.1/ADEL Security attribute based access control

FDP_ACF.1.1/ADEL

The TSF shall enforce the ADEL access control SFP to objects based on the following:

[1] [assignment: list of failures/service discontinuities]

[2] [assignment: quantification]

Table 51

Subject/Object	Attributes
S.JCVM	Active Applets
S.JCRE	Selected Applet Context, Registered Applets, Resident Packages
O.CODE_PKG	Package AID, Dependent Package AID, Static References
O.APPLET	Applet Selection Status

FDP_ACF.1.2/ADEL

The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

In the context of this policy, an object O is reachable if and only one of the following conditions hold:

- (1) the owner of O is a registered applet instance A (O is reachable from A),
- (2) a static field of a resident package P contains a reference to O (O is reachable from P),
- (3) there exists a valid remote reference to O (O is remote reachable),
- (4) there exists an object O' that is reachable according to either (1) or (2) or (3) above and O' contains a reference to O (the reachability status of O is that of O').

The following access control rules determine when an operation among controlled subjects and objects is allowed by the policy:

- R.JAVA.14 ([JCRE3] §11.3.4.1, Applet Instance Deletion): S.ADEL may perform OP.DELETE_APPLET upon an O.APPLET only if,
  - (1) S.ADEL is currently selected,
  - (2) there is no instance in the context of O.APPLET that is active in any logical channel and
  - (3) there is no O.JAVAOBJECT owned by O.APPLET such that either O.JAVAOBJECT is reachable from an applet instance distinct from O.APPLET, or O.JAVAOBJECT is reachable from a package P, or ([JCAPI3] §8.5) O.JAVAOBJECT is remote reachable.
- R.JAVA.15 ([JCRE3] §11.3.4.1, Multiple Applet Instance Deletion): S.ADEL may perform OP.DELETE_APPLET upon several O.APPLET only if,
  - (1) S.ADEL is currently selected,
  - (2) there is no instance of any of the O.APPLET being deleted that is active in any logical channel and
  - (3) there is no O.JAVAOBJECT owned by any of the O.APPLET being deleted such that either O.JAVAOBJECT is reachable from an applet instance distinct from any of those O.APPLET, or O.JAVAOBJECT is reachable from a package P, or ([JCRE3] §8.5) O.JAVAOBJECT is remote reachable.
- R.JAVA.16 ([JCRE3] §11.3.4.2, Applet/Library Package Deletion): S.ADEL may perform OP.DELETE_PCKG upon an O.CODE_PKG only if,
  - (1) S.ADEL is currently selected,
  - (2) no reachable O.JAVAOBJECT, from a package distinct from O.CODE_PKG that is an instance of a class that belongs to O.CODE_PKG, exists on the card and
  - (3) there is no resident package on the card that depends on O.CODE_PKG.
- R.JAVA.17 ([JCRE3] §11.3.4.3, Applet Package and Contained Instances Deletion): S.ADEL may perform OP.DELETE_PCKG_APPLET upon an O.CODE_PKG only if,
  - (1) S.ADEL is currently selected,

- (2) no reachable O.JAVAOBJECT, from a package distinct from O.CODE_PKG, which is an instance of a class that belongs to O.CODE_PKG exists on the card,
- (3) there is no package loaded on the card that depends on O.CODE_PKG, and
- (4) for every O.APPLET of those being deleted it holds that: (i) there is no instance in the context of O.APPLET that is active in any logical channel and (ii) there is no O.JAVAOBJECT owned by O.APPLET such that either O.JAVAOBJECT is reachable from an applet instance not being deleted, or O.JAVAOBJECT is reachable from a package not being deleted, or ([JCRE3] §8.5) O.JAVAOBJECT is remote reachable.

FDP_ACF.1.3/ADEL

The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: none.

FDP_ACF.1.4/ADEL [Editorially Refined]

The TSF shall explicitly deny access of any subject but S.ADEL to O.CODE_PKG or O.APPLET for the purpose of deleting them from the card.

7.2.3.3 FDP_RIP.1/ADEL Subset residual information protection

FDP_RIP.1.1/ADEL

The TSF shall ensure that any previous information content of a resource is made unavailable upon the deallocation of the resource from the following objects: applet instances and/or packages when one of the deletion operations in FDP_ACC.2.1/ADEL is performed on them.

7.2.3.4 FMT_MSA.1/ADEL Management of security attributes

FMT_MSA.1.1/ADEL

The TSF shall enforce the ADEL access control SFP to restrict the ability to modify the security attributes Registered Applets and Resident Packages to the Java Card RE.

7.2.3.5 FMT_MSA.3/ADEL Static attribute initialisation

FMT_MSA.3.1/ADEL

The TSF shall enforce the ADEL access control SFP to provide restrictive default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/ADEL

The TSF shall allow the following role(s): none, to specify alternative initial values to override the default values when an object or information is created.

7.2.3.6 FMT_SMF.1/ADEL Specification of Management Functions

FMT_SMF.1.1/ADEL

The TSF shall be capable of performing the following management functions: modify the list of registered applets' AIDs and the Resident Packages.

7.2.3.7 FMT_SMR.1/ADEL Security roles

FMT_SMR.1.1/ADEL

The TSF shall maintain the roles: applet deletion manager.

FMT_SMR.1.2/ADEL

The TSF shall be able to associate users with roles.

7.2.3.8 FPT_FLS.1/ADEL Failure with preservation of secure state

FPT_FLS.1.1/ADEL

The TSF shall preserve a secure state when the following types of failures occur: the applet deletion manager fails to delete a package/applet as described in [JCRE3]§11.3.4.

7.2.4 OdelG security functional requirements

The following requirements concern the object deletion mechanism. This mechanism is triggered by the applet that owns the deleted objects by invoking a specific API method.

7.2.4.1 FDP_RIP.1/ODEL Subset residual information protection

FDP_RIP.1.1/ODEL

The TSF shall ensure that any previous information content of a resource is made unavailable upon the deallocation of the resource from the following objects: the objects owned by the context of an applet instance which triggered the execution of the method javacard.framework.JCSystem.requestObjectDeletion().

7.2.4.2 FPT_FLS.1/ODEL Failure with preservation of secure state

FPT_FLS.1.1/ODEL

The TSF shall preserve a secure state when the following types of failures occur: the object deletion functions fail to delete all the unreferenced objects owned by the applet that requested the execution of the method.

7.2.5 CarG security functional requirements

This group includes requirements for preventing the installation of packages that has not been bytecode verified, or that has been modified after bytecode verification.

All SFRs of this group have been refined or remapped in group CMGRG, which specialises the SFRs from CarG to more specific SFRs of GlobalPlatform functionality.

Table 52 Refinement/Remapping of CarG SFRs

SFR from CargG	Corresponding SFR in CMGRG	Refinement/Remapping
FCO_NRO.2/CM Enforced proof of origin	FCO_NRO.2/SC Enforced proof of origin	Refinement
FDP_IFC.2/CM Complete information flow control	FDP_IFC.2/SC Complete information flow control	Refinement
FDP_UIT.1/CM Data exchange integrity	FDP_UIT.1/CCM Data exchange integrity	Remapping
FIA_UID.1/CM Timing of identification	FIA_UID.1/SC Timing of identification	Refinement
FMT_MSA.1/CM Management of security attributes	FMT_MSA.1/SC Management of security attributes	Remapping

SFR from CargG	Corresponding SFR in CMGRG	Refinement/Remapping
FMT_MSA.3/CM Static attribute initialisation	FMT_MSA.3/SC Static attribute initialisation	Remapping
FMT_SMF.1/CM Specification of Management functions	FMT_SMF.1/SC Specification of Management Functions	Remapping
FMT_SMR.1/CM Security roles	FMT_SMR.1/SD Security roles	Remapping
FTP_ITC.1/CM Inter-TSF trusted channel	FTP_ITC.1/SC Inter-TSF trusted channel	Remapping
FDP_IFF.1/CM Simple security attributes	FDP_IFF.1/SC Simple security attributes	Refinement

7.2.6 Card manager (CMGRG)

7.2.6.1 FDP_UIT.1/CCM Data exchange integrity

FDP_UIT.1.1/CCM

The TSF shall enforce the Secure Channel Protocol information flow control policy[1] to receive[2] user data in a manner protected from modification, deletion, insertion and replay[3] errors.

FDP_UIT.1.2/CCM

The TSF shall be able to determine on receipt of user data, whether modification, deletion, insertion, replay[4] has occurred.

7.2.6.2 FDP_ROL.1/CCM Basic rollback

FDP_ROL.1.1/CCM

The TSF shall enforce Security Domain access control policy[5] to permit the rollback of the installation operation[6] on the executable files and application instances[7].

FDP_ROL.1.2/CCM

The TSF shall permit operations to be rolled back within the bounds of the commit capacity (see [JCRE3] ch.7.8)[8].

7.2.6.3 FDP_ITC.2/CCM Import of user data with security attributes

FDP_ITC.2.1/CCM

[1] [assignment: access control SFP(s) and/or information flow control SFP(s)]

[2] [selection: transmit, receive]

[3] [selection: modification, deletion, insertion, replay]

[4] [selection: modification, deletion, insertion, replay]

[5] [assignment: access control SFP(s) and/or information flow control SFP(s)]

[6] [assignment: list of operations]

[7] [assignment: information and/or list of objects]

[8] [assignment: boundary limit to which rollback may be performed]

The TSF shall enforce the Firewall access control policy, the Security Domain access control policy and the Secure Channel Protocol information flow policy[1] when importing user data, controlled under the SFP, from outside of the TOE.

FDP_ITC.2.2/CCM

The TSF shall use the security attributes associated with the imported user data.

FDP_ITC.2.3/CCM

The TSF shall ensure that the protocol used provides for the unambiguous association between the security attributes and the user data received.

FDP_ITC.2.4/CCM

The TSF shall ensure that interpretation of the security attributes of the imported user data is as intended by the source of the user data.

FDP_ITC.2.5/CCM

The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: Package loading is allowed only if, for each dependent package, its AID attribute is equal to a resident package AID attribute, the major (minor) Version attribute associated to the dependent package is lesser than or equal to the major (minor) Version attribute associated to the resident package ([JCVM3] §4.5.2)[2].

7.2.6.4 FPT_FLS.1/CCM Failure with preservation of secure state

FPT_FLS.1.1/CCM

The TSF shall preserve a secure state when the following types of failures occur: the Security Domain fails to load/install an Executable File / application instance as described in [JCRE3] Section 11.1.5[3].

7.2.6.5 FCS_COP.1/DAP Cryptographic operation

FCS_COP.1.1/DAP

The TSF shall perform verification of the DAP signature attached to Executable Load Applications[4] in accordance with a specified cryptographic algorithm

- PKC Scheme: SHA-1 hash and [PKCS v2.2] RSA signature[5]
- PKC Scheme: RSA key of minimum length 1024 bits[6]

that meet the following

- Sections C.1.2 and C.6 of [GP v23]

[1] [assignment: access control SFP(s) and/or information flow control SFP(s)]

[2] [assignment: additional importation control rules]

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

[4] [assignment: list of cryptographic operations]

[5] [assignment: cryptographic algorithm]

[6] [assignment: cryptographic key sizes]

- PKC Scheme: SSA-PKCS1-v1_5 as defined in [PKCS v2.2][1]

7.2.6.6 FDP_ACC.1/SD Subset access control

FDP_ACC.1.1/SD

The TSF shall enforce the Security Domain access control policy[2] on:

- Subjects: S.INSTALLER, S.ADEL, S.CAD (from [JC PP]) and S.SD
- Objects: Delegation Token, DAP Block and Load File
- Operations: GlobalPlatform's card content management APDU commands and API methods.
- proprietary administration commands listed in [ADMIN][3]
-

7.2.6.7 FDP_ACF.1/SD Security attribute based access control

FDP_ACF.1.1/SD

The TSF shall enforce the Security Domain access control policy[4] to objects based on the following:

- Subjects:
  - S.INSTALLER, defined in [JC PP] and represented by the GlobalPlatform Environment (OPEN) on the card, the Card Life Cycle attributes (defined in Section 5.1.1 of [GP v23]);
  - S.ADEL, also defined in [JC PP] and represented by the GlobalPlatform Environment (OPEN) on the card;
  - S.SD receiving the Card Content Management commands (through APDUs or APIs) with a set of privileges (defined in Section 6.6.1 of [GP v23]), a life-cycle status (defined in Section 5.3.2 of [GP v23]) and a Secure Communication Security level (defined in Section 10.6 of [GP v23]);
  - S.CAD, defined in [JC PP], the off-card entity that communicates with the S.INSTALLER through S.SD;
- Objects:
  - The Delegation Token, in case of Delegated Management operations, with the attributes Present or Not Present;
  - The DAP Block, in case of application loading, with the attributes Present or Not Present;
  - The Load File or Executable File, in case of application loading, installation, extradition or registry update, with a set of intended privileges and its targeted associated SD AID.5
  -

FDP_ACF.1.2/SD

The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:[6]

Runtime behavior rules defined by GlobalPlatform for:

[1] [assignment: list of standards]

[2] [assignment: access control SFP]

[3] [assignment: list of subjects, objects, and operations among subjects and objects covered by the SFP]

[4] [assignment: access control SFP]

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

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

- loading (Section 9.3.5 of [GP v23])
- installation (Section 9.3.6 of [GP v23])
- extradition (Section 9.4.1 of [GP v23])
- registry update (Section 9.4.2 of [GP v23] and Section 2.8 of [ADMIN])
- content removal (Section 9.5 of [GP v23])

FDP_ACF.1.3/SD

The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: Proprietary runtime behavior rules defined by [ADMIN] for[1]:

- reinitialize OS (Section 2.10.3.1 of [ADMIN])
- config OS (Section 2.10.3.4 of [ADMIN]).

Note: (Proprietary registry update): The product supports a mode called Static Mode in which INSTALL [for LOAD] and LOAD command processing is blocked. Static Mode can be enabled using INSTALL [for registry update] command. Refer to [ADMIN] for more details.

Note: Native mode (section 2.9 of [ADMIN]) can be activated by a Config OS command.

FDP_ACF.1.4/SD

The TSF shall explicitly deny access of subjects to objects based on the following additional rules: when at least one of the rules defined by GlobalPlatform does not hold[2].

7.2.6.8 FMT_MSA.1/SD Management of security attributes

FMT_MSA.1.1/SD

The TSF shall enforce the Security Domain access control policy[3] to restrict the ability to modify[4] the security attributes

- life-cycle status
- card life cycle
- privileges
- security level[5]

to the Security Domain and the application instance itself[6].

7.2.6.9 FMT_MSA.3/SD Static attribute initialisation

FMT_MSA.3.1/SD

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

[2] [assignment: rules, based on security attributes, that explicitly deny access of subjects to objects].

[3] [assignment: access control SFP(s), information flow control SFP(s)]

[4] [selection: change_default, query, modify, delete, [assignment: other operations]]

[5] [assignment: list of security attributes]

[6] [assignment: the authorised identified roles]

The TSF shall enforce the security domain access control policy[1] to provide restrictive[2] default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/SD

The TSF shall allow the card issuer, application provider[3] to specify alternative initial values to override the default values when an object or information is created.

Refinement: Alternative initial values shall be at least as restrictive as the default values defined in FMT_MSA.3.1.

7.2.6.10 FMT_SMF.1/SD Specification of Management Functions

FMT_SMF.1.1/SD

The TSF shall be capable of performing the following management functions:

- card locking (Section 9.6.3 of [GP v23])
- application locking and unlocking (Section 9.6.2 of [GP v23])
- card termination (Section 9.6.4 of [GP v23])
- card status interrogation (Section 9.6.6 of [GP v23])
- application status interrogation (Section 9.6.5 of [GP v23]).

7.2.6.11 FMT_SMR.1/SD Security roles

FMT_SMR.1.1/SD

The TSF shall maintain the roles.

- Card Issuer (ISD Authenticated)
- Application Provider (SSD Authenticated)
- Application Instance[4]

FMT_SMR.1.2/SD

The TSF shall be able to associate users with roles.

7.2.6.12 FTP_ITC.1/SC Inter-TSF trusted channel

FTP_ITC.1.1/SC

The TSF shall provide a communication channel between itself and another trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure.

[1] [assignment: access control SFP, information flow control SFP]

[2] [selection, choose one of: restrictive, permissive, [assignment: other property]]

[3] [assignment: the authorised identified roles]

[4] [assignment: the authorised identified roles]

FTP_ITC.1.2/SC

The TSF shall permit another trusted IT product[1] to initiate communication via the trusted channel.

FTP_ITC.1.3/SC

The TSF shall initiate communication via the trusted channel for all card management functions[2]:

- loading;
- installation;
- extradition;
- registry update;
- SD personalization;
- SD update.

7.2.6.13 FCO_NRO.2/SC Enforced proof of origin

FCO_NRO.2.1/SC

The TSF shall enforce the generation of evidence of origin for transmitted Executable load files[3] at all times.

FCO_NRO.2.2/SC

The TSF shall be able to relate the DAP signature block[4] of the originator of the information, and the executable load file content[5] of the information to which the evidence applies.

FCO_NRO.2.3/SC

The TSF shall provide a capability to verify the evidence of origin of information to recipient[6] at the time the Executable load files are received as no evidence is kept on the card for future verification[7].

Note: DAP verification is one method to provide evidence for the originator of the executable load file but it is not mandatory. Such evidence could also be provided by the secure channel and an appropriate organizational security policy.

7.2.6.14 FDP_IFC.2/SC Complete information flow control

FDP_IFC.2.1/SC

The TSF shall enforce the Secure Channel Protocol information flow control policy[8] on

- the subjects S.CAD and S.SD, involved in the exchange of messages between the Java Card and the CAD through a potentially unsafe communication channel

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

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

[3] [assignment: list of information types]

[4] [assignment: list of attributes]

[5] [assignment: list of information fields]

[6] [selection: originator, recipient, [assignment: list of third parties]]

[7] [assignment: limitations on the evidence of origin].

[8] [assignment: information flow control SFP]

- the information controlled by this policy is the card content management command, including personalization commands, in the APDUs sent to the card and their associated responses returned to the CAD.[1]

and all operations that cause that information to flow to and from subjects covered by the SFP.

FDP_IFC.2.2/SC

The TSF shall ensure that all operations that cause any information in the TOE to flow to and from any subject in the TOE are covered by an information flow control SFP.

Note: This SFR is interpreted as a refinement of the SFR FDP_IFC.2/CM from [JC PP]. The subject S.INSTALLER is generalized to S.SD, which is the agent for any management command. The subject S.BCV from SFR FDP_IFC.2/CM has been removed because the byte code verifier belongs to the operational environment and is therefor not a valid subject.

7.2.6.15 FDP_IFF.1/SC Simple security attributes

FDP_IFF.1.1/SC

The TSF shall enforce the Secure Channel Protocol information flow control policy[2] based on the following types of subject and information security attributes:

- Subjects:
  - S.SD receiving the Card Content Management commands (through APDUs or APIs). This subject can be the ISD, an APSD or a CASD.
  - S.CAD the off-card entity that communicates with the S.SD.
- Information:
  - load file, in case of application loading;
  - applications or SD privileges, in case of application installation or registry update;
  - personalization keys and/or certificates, in case of application or SD personnalization.

FDP_IFF.1.2/SC

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

Runtime behavior rules defined by GlobalPlatform for:

- loading (Section 9.3.5 of [GP v23]);
- installation (Section 9.3.6 of [GP v23]);
- extradition (Section 9.4.1 of [GP v23]);
- registry update (Section 9.4.2 of [GP v23]);
- content removal (Section 9.5 of [GP v23]).[3]

FDP_IFF.1.3/SC

[1] [assignment: list of subjects and information]

[2] [assignment: information flow control SFP]

[3] [assignment: for each operation, the security attribute-based relationship that must hold between subject and information security attributes]

The TSF shall enforce the none[1].

FDP_IFF.1.4/SC

The TSF shall explicitly authorise an information flow based on the following rules: none[2].

FDP_IFF.1.5/SC

The TSF shall explicitly deny an information flow based on the following rules: When none of the conditions listed in the element FDP_IFF.1.4 of this component hold and at least one of those listed in the element FDP_IFF.1.2 does not hold[3]

7.2.6.16 FMT_MSA.1/SC Management of security attributes

FMT_MSA.1.1/SC

The TSF shall enforce the Secure Channel Protocol (SCP) information flow control policy[4] to restrict the ability to modify[5] the security attributes SCP keys[6] to card issuer and application provider[7].

7.2.6.17 FMT_MSA.3/SC Static attribute initialisation

FMT_MSA.3.1/SC

The TSF shall enforce the Secure Channel Protocol (SCP) information flow control policy[8] to provide restrictive[9] default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/SC

The TSF shall allow the (Card) Issuer and Application Provider[10] to specify alternative initial values to override the default values when an object or information is created.

7.2.6.18 FMT_SMF.1/SC Specification of Management Functions

FMT_SMF.1.1/SC

The TSF shall be capable of performing the following management functions:

- Management functions specified in GlobalPlatform specifications [GP v23]:
- loading (Section 9.3.5 of [GP v23]);

[1] [assignment: additional information flow control SFP rules]
[2] [assignment: rules, based on security attributes, that explicitly authorise information flows]
[3] [assignment: rules, based on security attributes, that explicitly deny information flows]
[4] [assignment: access control SFP(s), information flow control SFP(s)]
[5] [selection: change_default, query, modify, delete, [assignment: other operations]]
[6] [assignment: list of security attributes]
[7] [assignment: the authorised identified roles]
[8] [assignment: access control SFP, information flow control SFP]
[9] [selection, choose one of: restrictive, permissive, [assignment: other property]]
[10] [assignment: the authorised identified roles]

- installation (Section 9.3.6 of [GP v23]);
- extradition (Section 9.4.1 of [GP v23]);
- registry update (Section 9.4.2 of [GP v23]);[1]

7.2.6.19 FIA_UID.1/SC Timing of identification

FIA_UID.1.1/SC

The TSF shall allow

- application selection;
- initializing a secure channel with the card;
- open a supplementary logical channel
- verify password[2]

on behalf of the user to be performed before the user is identified.

> *Note: "Verify password" is a proprietary Password authentication described in [ADMIN].*

FIA_UID.1.2/SC

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

7.2.6.20 FIA_UAU.1/SC Timing of authentication

FIA_UAU.1.1/SC

The TSF shall allow the TSF mediated actions listed in FIA_UID.1/SC[3] on behalf of the user to be performed before the user is authenticated.

FIA_UAU.1.2/SC

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

7.2.6.21 FIA_UAU.4/SC Single-use authentication mechanisms

FIA_UAU.4.1/SC

The TSF shall prevent reuse of authentication data related to the authentication mechanism used to open a secure communication channel with the card[4].

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

[2] [assignment: list of TSF-mediated actions]

[3] [assignment: list of TSF mediated actions]

[4] [assignment: identified authentication mechanism(s)].

7.2.7 SCPG security functional requirements

The group SCPG contains the security requirements from the underlying platform. The following SFRs are taken from [ST_IC]. Their exact definition will not be repeated here. For details, please see [ST_IC].

7.2.7.1 FPT_PHP.3 Resistance to physical attacks

FPT_PHP.3.1

The TSF shall resist physical manipulation and probing[1] to the all TSFs[2] by responding automatically such that the SFRs are always enforced.

7.2.7.2 FPT_TST.1 TSF testing

FPT_TST.1.1

The TSF shall run a suite of self tests during initial start-up[3] to demonstrate the correct operation of the TSF[4].

FPT_TST.1.2

The TSF shall provide authorised users with the capability to verify the integrity of the TSF data[5].

FPT_TST.1.3

The TSF shall provide authorised users with the capability to verify the integrity of TSF[6].

Note: "During Initial startup" means during each power up.

Note: The automatic startup tests demonstrate the correct operation of the alarm lines and/or the environmental sensor mechanisms.

7.2.7.3 FCS_RNG.1 Random number generation (Class PTG.3)

FCS_RNG.1.1

The TSF shall provide a hybrid physical random number generator that implements:

(PTG.3.1) A total failure test detects a total failure of entropy source immediately when the RNG has started. When a total failure has been detected no random numbers will be output.

(PTG.3.2) 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[7].

[1] [assignment: physical tampering scenarios]

[2] [assignment: list of TSF devices/elements]

[3] [selection: during initial start-up, periodically during normal operation, at the request of the authorised user, at the conditions[assignment: conditions under which self test should occur]]

[4] [selection: [assignment: parts of TSF], the TSF]

[5] [selection: [assignment: parts of TSF data], TSF data]

[6] [selection: [assignment: parts of TSF], TSF]

[7] [selection: 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, generates the internal random numbers with a post-processing algorithm of class DRG.3 as long as its internal state entropy guarantees the claimed output entropy]

(PTG.3.3) The online test shall detect non-tolerable statistical defects of the raw random number sequence (i) immediately when the RNG is started, and (ii) while the RNG is being operated. The TSF must not output any random numbers before the power-up online test and the seeding of the DRG.3 postprocessing algorithm have been finished successfully or when a defect has been detected.

(PTG.3.4) The online test procedure shall be effective to detect non-tolerable weaknesses of the random numbers soon.

(PTG.3.5) The online test procedure checks the raw random number sequence. It is triggered continuously[1]. The online test is suitable for detecting nontolerable statistical defects of the statistical properties of the raw random numbers within an acceptable period of time.

(PTG.3.6) The algorithmic post-processing algorithm belongs to Class DRG.3 with cryptographic state transition function and cryptographic output function, and the output data rate of the post-processing algorithm shall not exceed its input data rate.

FCS_RNG.1.2

The TSF shall provide octets of bits[2] that meet:

(PTG.3.7) Statistical test suites cannot practically distinguish the internal random numbers from output sequences of an ideal RNG. The internal random numbers must pass test procedure A.

(PTG.3.8) The internal random numbers shall use PTRNG of class PTG.2 as random source for the post-processing[3].

7.3 Security requirements rationale

7.3.1 Security functional requirements rationale

The following table provides a mapping form Objectives to SFRs. This mapping is an adapted version form the mapping provided in [JC PP].

The underlined SFRs are new in this ST. They are either refinements/remappings form the group CARG (see chapter 7.2.5) or additional SFRs defined in CMGRG (chapter 7.2.6) and SCPG (chapter 7.2.7).

The underlined Objectives are new in this ST. They have been introduced due to the new groups CMGRG (chapter 7.2.6) and SCPG (chapter 7.2.7).

Table 53 **Mapping Security Objectives to SFRs**

Security Objective	SFRs
O.SID	As in [JC PP] with remapping/refinement according to ch. 7.2.5. Please note that FMT_MSA.1/REM_REFS and FMT_MSA.1/EXPORT have been wrongly included in [JC PP] and must be removed.
O.FIREWALL	As in [JC PP] with remapping/refinement according to ch. 7.2.5.
O.GLOBAL_ARRAYS_CONFID	As in [JC PP]
O.GLOBAL_ARRAYS_INTEG	As in [JC PP]

[1] [selection: externally, at regular intervals, continuously, upon specified internal events]

[2] [selection: bits, octets of bits, numbers [assignment: format of the numbers]]

[3] [selection: use PTRNG of class PTG.2 as random source for the post-processing, have [assignment: work factor], require [assignment: guess work]]

Security Objective	SFRs
O.NATIVE	As in [JC PP]
O.OPERATE	As in [JC PP]
O.REALLOCATION	As in [JC PP]
O.RESOURCES	As in [JC PP] with remapping/refinement according to ch. 7.2.5.
O.ALARM	As in [JC PP]
O.CIPHER	As in [JC PP]
O.KEY-MNGT	As in [JC PP]
O.PIN-MNGT	As in [JC PP]
O.TRANSACTION	As in [JC PP]
O.OBJ-DELETION	As in [JC PP]
O.DELETION	As in [JC PP]
O.LOAD	As in [JC PP] with remapping/refinement according to ch. 7.2.5.
O.INSTALL	As in [JC PP]
O.COMMUNICATION	FMT_SMR.1/SD, FMT_MSA.1/SC, FMT_MSA.3/SC, FIA_UID.1/SC, FTP_ITC.1/SC, FDP_IFC.2/SC, FDP_IFF.1/SC, FIA_UAU.1/SC, FMT_SMF.1/SC
O.RNG	FCS_RNG.1
O.CARD-MANAGEMENT	FDP_ACC.1/SD, FMT_MSA.1/SD, FMT_MSA.3/SD, FMT_SMF.1/SD, FMT_SMR.1/SD, FDP_UIT.1/CCM, FDP_ROL.1/CCM, FDP_ITC.2/CCM, FMT_MSA.1/SC, FMT_MSA.3/SC, FMT_SMF.1/SC, FIA_UAU.1/SC, FTP_ITC.1/SC, FCO_NRO.2/SC, FDP_IFC.2/SC, FDP_IFF.1/SC, FIA_UID.1/SC, FIA_UAU.4/SC, FDP_ACF.1/SD, FPT_FLS.1/CCM
O.SCP.SUPPORT	FCS_COP.1 , FDP_ROL.1/FIREWALL
O.SCP.IC	FPT_PHP.3, FPT_TST.1
O.SCP.RECOVERY	FPT_FLS.1, FAU_ARP.1

7.3.1.1 O.SID

The rationale for the SFR mapping is stated in [JC PP]. The underlined SFRs are refinements of the corresponding SFRs from the CARG group (see chapter 7.2.5).

7.3.1.2 O.FIREWALL

The rationale for the SFR mapping is stated in [JC PP]. The underlined SFRs are refinements of the corresponding SFRs from the CARG group (see chapter 7.2.5).

7.3.1.3 O.GLOBAL_ARRAYS_CONFID

The rationale for the SFR mapping is stated in [JC PP].

7.3.1.4 O.NATIVE

The rationale for the SFR mapping is stated in [JC PP].

7.3.1.5 O.OPERATE

The rationale for the SFR mapping is stated in [JC PP].

7.3.1.6 O.REALLOCATION

The rationale for the SFR mapping is stated in [JC PP].

7.3.1.7 O.RESOURCES

The rationale for the SFR mapping is stated in [JC PP]. The underlined SFRs are refinements of the corresponding SFRs from the CARG group (see chapter 7.2.5).

7.3.1.8 O.ALARM

The rationale for the SFR mapping is stated in [JC PP].

7.3.1.9 O.CIPHER

The rationale for the SFR mapping is stated in [JC PP].

7.3.1.10 O.KEY-MNGT

The rationale for the SFR mapping is stated in [JC PP].

7.3.1.11 O.PIN-MNGT

The rationale for the SFR mapping is stated in [JC PP].

7.3.1.12 O.TRANSACTION

The rationale for the SFR mapping is stated in [JC PP].

7.3.1.13 O.OBJ-DELETION

The rationale for the SFR mapping is stated in [JC PP].

7.3.1.14 O.DELETION

The rationale for the SFR mapping is stated in [JC PP].

7.3.1.15 O.LOAD

The rationale for the SFR mapping is stated in [JC PP]. The underlined SFRs are refinements of the corresponding SFRs from the CARG group (see chapter 7.2.5).

7.3.1.16 O.INSTALL

The rationale for the SFR mapping is stated in [JC PP].

7.3.1.17 O.COMMUNICATION

This objective has been introduced in this Security Target. It is covered by the following security functional requirements:

- FTP_ITC.1/SC which ensures the origin, integrity and confidentiality of card administration commands.
- FMT_SMR.1/SD specifies the authorized identified roles enabling to send and authenticate card management commands.
- FDP_IFC.2/SC and FDP_IFF.1/SC enforces the Secure Channel Protocol information flow control policy to ensure the origin, integrity and confidentiality of administration requests.
- FMT_MSA.1/SC and FMT_MSA.3/SC covers indirectly this security objective by specifying security attributes enabling to authenticate card management requests and to guarantee the integrity and confidentiality of card management requests.
- FIA_UID.1/SC and FIA_UAU.1/SC specify the actions that can be performed before authenticating the origin of the APDU commands that the card receives.
- FMT_SMF.1/SC specifies the actions activating the authentication, confidentiality and integrity check on the card management commands.

The security functional requirement FCS_COP.1 defined in [JCRE] supports also this security objective by specifying secure cryptographic algorithm that shall be used to determine the origin of the card management commands.

7.3.1.18 O.RNG

This objective is covered by FCS_RNG.1.

7.3.1.19 O.CARD-MANAGEMENT

This objective has been introduced in this Security Target. It has been remapped form the corresponding objective for the operational environment O.CARD-MANAGEMENT.

The security objective O.CARD-MANAGEMENT is met by the following SFRs:

- FDP_UIT.1/CCM enforces the Secure Channel Protocol information flow control policy and the Security Domain access control policy to ensure the integrity of card management operations.
- FDP_ROL.1/CCM ensures that card management operations may be cleanly aborted.
- FDP_ITC.2/CCM enforces the Firewall access control policy and the Secure Channel Protocol information flow policy when importing card management data.
- FPT_FLS.1/CCM preserves a secure state when failures occur.
- All SFRs related to Security Domains (FDP_ACC.1/SD, FDP_ACF.1/SD, FMT_MSA.1/SD, FMT_MSA.3/SD, FMT_SMF.1/SD, FMT_SMR.1/SD) cover this security objective by enforcing a Security Domain access control policy (rules and restrictions) that ensures a secure card content management.
- All SFRs related to the secure channel (FMT_MSA.1/SC, FMT_MSA.3/SC, FMT_SMF.1/SC, FIA_UAU.1/SC, FTP_ITC.1/SC, FCO_NRO.2/SC, FDP_IFC.2/SC, FDP_IFF.1/SC, FIA_UID.1/SC, FIA_UAU.4/SC) support this security objective by enforcing Secure Channel Protocol information flow control policy that ensures the integrity and the authenticity of card management operations.
-

7.3.1.20 O.SCP.RECOVERY

This objective is covered as follows:

- FAU_ARP.1 provides one aspect which refers to card tearing and power failure.
- FPT_FLS.1 states that the TOE shall preserve a secure state if an event according to FAU_ARP.1 occurs

7.3.1.21 O.SCP.SUPPORT

This objective is covered as follows:

- FCS_COP.1 provides maps to low-level-cryptographic support
- FDP_ROL.1/FIREWAL maps to low-level transaction mechanism

Note: As used in CC version 3, non-bypassability will be treated in ADV_ARC

7.3.1.22 O.SCP.IC

This objective is covered by FPT_PHP.3.

Table 54 Mapping SFR to Security Objectives

SFR	Security Objectives
FDP_ACC.2/FIREWALL	as in [JC PP]
FDP_ACF.1/FIREWALL	as in [JC PP]
FDP_IFC.1/JCVM	as in [JC PP]
FDP_IFF.1/JCVM	as in [JC PP]
FDP_RIP.1/OBJECTS	as in [JC PP]
FMT_MSA.1/JCRE	as in [JC PP]
FMT_MSA.1/JCVM	as in [JC PP]
FMT_MSA.2/FIREWALL_JCVM	as in [JC PP]
FMT_MSA.3/FIREWALL	as in [JC PP]
FMT_MSA.3/JCVM	as in [JC PP]
FMT_SMF.1	as in [JC PP]
FMT_SMR.1	as in [JC PP]
FCS_CKM.1	as in [JC PP]
FCS_CKM.2	as in [JC PP]
FCS_CKM.3	as in [JC PP]
FCS_CKM.4	as in [JC PP]
FCS_COP.1	as in [JC PP]
FDP_RIP.1/ABORT	as in [JC PP]
FDP_RIP.1/APDU	as in [JC PP]
FDP_RIP.1/bArray	as in [JC PP]
FDP_RIP.1/KEYS	as in [JC PP]
FDP_RIP.1/TRANSIENT	as in [JC PP]
FDP_ROL.1/FIREWALL	as in [JC PP]
FAU_ARP.1	as in [JC PP]
FDP_SDI.2	as in [JC PP]
FPR_UNO.1	as in [JC PP]
FPT_FLS.1	as in [JC PP]
FPT_TDC.1	as in [JC PP]

SFR	Security Objectives
FIA_ATD.1/AID	as in [JC PP]
FIA_UID.2/AID	as in [JC PP]
FIA_USB.1/AID	as in [JC PP]
FMT_MTD.1/JCRE	as in [JC PP]
FMT_MTD.3/JCRE	as in [JC PP]
FDP_ITC.2/Installer	as in [JC PP]
FMT_SMR.1/Installer	as in [JC PP]
FPT_RCV.3/Installer	as in [JC PP]
FPT_FLS.1/Installer	as in [JC PP]
FDP_ACC.2/ADEL	as in [JC PP]
FDP_ACF.1/ADEL	as in [JC PP]
FDP_RIP.1/ADEL	as in [JC PP]
FMT_MSA.1/ADEL	as in [JC PP]
FMT_MSA.3/ADEL	as in [JC PP]
FMT_SMF.1/ADEL	as in [JC PP]
FMT_SMR.1/ADEL	as in [JC PP]
FPT_FLS.1/ADEL	as in [JC PP]
FDP_RIP.1/ODEL	as in [JC PP]
FPT_FLS.1/ODEL	as in [JC PP]
FDP_UIT.1/CCM	O.LOAD, O.CARD-MANAGEMENT
FDP_ROL.1/CCM	O.OPERATE, O.RESOURCES, O.DELETION, O.INSTALL, O.RECOVERY, O.CARD-MANAGEMENT
FDP_ITC.2/CCM	O.SID, O.FIREWALL, O.OPERATE, O.INSTALL, O.CARD-MANAGEMENT
FDP_ITC.2/CCM	O.CARD-MANAGEMENT
FDP_ACC.1/SD	O.CARD-MANAGEMENT
FDP_ACF.1/SD	O.CARD-MANAGEMENT
FMT_MSA.1/SD	O.CARD-MANAGEMENT
FMT_MSA.3/SD	O.CARD-MANAGEMENT
FMT_SMF.1/SD	O.CARD-MANAGEMENT
FMT_SMR.1/SD	O.FIREWALL, O.RESOURCES, O.CARD-MANAGEMENT, O.COMMUNICATION
FTP_ITC.1/SC	O.LOAD, O.CARD-MANAGEMENT, O.COMMUNICATION
FCO_NRO.2/SC	O.LOAD, O.CARD-MANAGEMENT
FDP_IFC.2/SC	O.LOAD, O.CARD-MANAGEMENT, O.COMMUNICATION,
FDP_IFF.1/SC	O.LOAD, O.CARD-MANAGEMENT, O.COMMUNICATION,
FMT_MSA.1/SC	O.SID, O.FIREWALL, O.CARD-MANAGEMENT, O.COMMUNICATION,
FMT_MSA.3/SC	O.SID, O.FIREWALL, O.CARD-MANAGEMENT, O.COMMUNICATION,
FMT_SMF.1/SC	O.SID, O.FIREWALL, O.RESOURCES, O.CARD-MANAGEMENT
FIA_UID.1/SC	O.COMMUNICATION, O.CARD-MANAGEMENT

SFR	Security Objectives
FIA_UAU.1/SC	O.COMMUNICATION, O.CARD-MANAGEMENT
FIA_UAU.4/SC	O.COMMUNICATION, O.CARD-MANAGEMENT
FCS_RNG.1	O.RNG
FPT_PHP.3	O.SCP.IC
FPT_TST.1	O.SCP.IC

7.3.2 Security assurance requirements rationale

The level EAL6+ has been chosen, because this Java Card platform will host applets with the highest security requirements (e.g. Government ID and signature applications).
The augmentation ALC_FLR.1 has been added to support systematic maintenance of the TOE over its complete life cycle.

7.3.2.1 ADV_SPM.1

The EAL6+ level contains the SAR ADV_SPM.1, which is called the Security Policy Model (SPM).
The details of the model are described in [SPM].

ADV_SPM.1.1D The developer shall provide a formal security policy model for the Firewall access control policy (ch. 7.2.1.1)[1].

ADV_SPM.1.2D For each policy covered by the formal security policy model, the model shall identify the relevant portions of the statement of SFRs that make up that policy.

ADV_SPM.1.3D The developer shall provide a formal proof of correspondence between the model and any formal functional specification.

ADV_SPM.1.4D The developer shall provide a demonstration of correspondence between the model and the functional specification.

7.4 Dependencies

7.4.1 SFR dependencies

Table 55 SFR Dependencies

Requirements	CC Dependencies	Satisfied Dependencies
FDP_ITC.2/Installer	as in [JC PP]	as in [JC PP]
FMT_SMR.1/Installer	as in [JC PP]	as in [JC PP]
FPT_FLS.1/Installer	as in [JC PP]	as in [JC PP]
FPT_RCV.3/Installer	as in [JC PP]	as in [JC PP]
FDP_ACC.2/ADEL	as in [JC PP]	as in [JC PP]
FDP_ACF.1/ADEL	as in [JC PP]	as in [JC PP]
FDP_RIP.1/ADEL	as in [JC PP]	as in [JC PP]

[1] [assignment: list of policies that are formally modelled]

Requirements	CC Dependencies	Satisfied Dependencies
FMT_MSA.1/ADEL	as in [JC PP]	as in [JC PP]
FMT_MSA.3/ADEL	as in [JC PP]	as in [JC PP]
FMT_SMF.1/ADEL	as in [JC PP]	as in [JC PP]
FMT_SMR.1/ADEL	as in [JC PP]	as in [JC PP]
FPT_FLS.1/ADEL	as in [JC PP]	as in [JC PP]
FDP_RIP.1/ODEL	as in [JC PP]	as in [JC PP]
FPT_FLS.1/ODEL	as in [JC PP]	as in [JC PP]
FDP_UIT.1/CCM	(FDP_ACC.1 or FDP_IFC.1) and (FTP_ITC.1 or FTP_TRP.1)	FDP_IFC.2/SC, FTP_ITC.2/CCM
FDP_ROL.1/CCM	(FDP_ACC.1 or FDP_IFC.1)	FDP_ACC.1/SD
FDP_ITC.2/CCM	(FDP_ACC.1 or FDP_IFC.1) and (FPT_TDC.1) and (FTP_ITC.1 or FTP_TRP.1)	FDP_ACC.1/SD, FTP_ITC.1/SC
FPT_FLS.1/CCM	No dependencies	
FCS_COP.1/DAP	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FDP_ITC.2/CCM
FDP_ACC.1/SD	(FDP_ACF.1)	FDP_ACF.1/SD
FDP_ACF.1/SD	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.1/SD, FMT_MSA.3/SD
FMT_MSA.1/SD	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.1/SD, FMT_SMF.1/SD, FMT_SMR.1/SD
FMT_MSA.3/SD	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/SD, FMT_SMR.1/SD
FMT_SMF.1/SD	No dependencies	
FMT_SMR.1/SD	(FIA_UID.1)	FIA_UID.1/SC
FTP_ITC.1/SC	No dependencies	
FCO_NRO.2/SC	(FIA_UID.1)	FIA_UID.1/SC
FDP_IFC.2/SC	(FDP_IFF.1)	FDP_IFF.1/SC
FDP_IFF.1/SC	(FDP_IFC.1) and (FMT_MSA.3)	FDP_IFC.2/SC, FMT_MSA.3/SC
FMT_MSA.1/SC	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.1/SD, FMT_SMR.1/SD, FMT_SMF.1/SC
FMT_MSA.3/SC	(FMT_MSA.1) and (FMT_SMR.1)	FMT_SMR.1/SD, FMT_MSA.1/SC
FMT_SMF.1/SC	No dependencies	
FIA_UID.1/SC	No dependencies	
FIA_UAU.1/SC	(FIA_UID.1)	FIA_UID.1/SC
FIA_UAU.4/SC	No dependencies	
FDP_ACC.2/FIREWALL	as in [JC PP]	as in [JC PP]
FDP_ACF.1/FIREWALL	as in [JC PP]	as in [JC PP]
FDP_IFC.1/JCVM	as in [JC PP]	as in [JC PP]
FDP_IFF.1/JCVM	as in [JC PP]	as in [JC PP]
FDP_RIP.1/OBJECTS	as in [JC PP]	as in [JC PP]

Requirements	CC Dependencies	Satisfied Dependencies
FMT_MSA.1/JCRE	as in [JC PP]	as in [JC PP]
FMT_MSA.1/JCVM	as in [JC PP]	as in [JC PP]
FMT_MSA.2/FIREWALL_JCVM	as in [JC PP]	as in [JC PP]
FMT_MSA.3/FIREWALL	as in [JC PP]	as in [JC PP]
FMT_MSA.3/JCVM	as in [JC PP]	as in [JC PP]
FMT_SMF.1	as in [JC PP]	as in [JC PP]
FMT_SMR.1	as in [JC PP]	as in [JC PP]
FCS_CKM.1	as in [JC PP]	as in [JC PP]
FCS_CKM.2	as in [JC PP]	as in [JC PP]
FCS_CKM.3	as in [JC PP]	as in [JC PP]
FCS_CKM.4	as in [JC PP]	as in [JC PP]
FCS_COP.1	as in [JC PP]	as in [JC PP]
FDP_RIP.1/ABORT	as in [JC PP]	as in [JC PP]
FDP_RIP.1/APDU	as in [JC PP]	as in [JC PP]
FDP_RIP.1/bArray	as in [JC PP]	as in [JC PP]
FDP_RIP.1/KEYS	as in [JC PP]	as in [JC PP]
FDP_RIP.1/TRANSIENT	as in [JC PP]	as in [JC PP]
FDP_ROL.1/FIREWALL	as in [JC PP]	as in [JC PP]
FAU_ARP.1	as in [JC PP]	as in [JC PP]
FDP_SDI.2	as in [JC PP]	as in [JC PP]
FPR_UNO.1	as in [JC PP]	as in [JC PP]
FPT_FLS.1	as in [JC PP]	as in [JC PP]
FPT_TDC.1	as in [JC PP]	as in [JC PP]
FIA_ATD.1/AID	as in [JC PP]	as in [JC PP]
FIA_UID.2/AID	as in [JC PP]	as in [JC PP]
FIA_USB.1/AID	as in [JC PP]	as in [JC PP]
FMT_MTD.1/JCRE	as in [JC PP]	as in [JC PP]
FMT_MTD.3/JCRE	as in [JC PP]	as in [JC PP]
FCS_RNG.1	No Dependencies	n.A.
FPT_PHP.3	No Dependencies	n.A.
FPT_TST.1	No dependencies	n.A.

Note: FCS_CKM.1 relates to all iterations defined in 7.2.1.13

Note: FCS_COP.1 relates to all iterations defined in 7.2.1.17

Note: The dependency FIA_UID.1 of FMT_SMR.1/Installer is unsupported. This ST does not require the identification of the “installer” since it can be considered as part of the TSF.

Note: The dependency FIA_UID.1 of FMT_SMR.1/ADEL is unsupported. This ST does not require the identification of the “deletion manager” since it can be considered as part of the TSF.

Note: The dependency FMT_SMF.1 of FMT_MSA.1/JCRE is unsupported. 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.

Note: The dependency FAU_SAA.1 of FAU_ARP.1 is unsupported. 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 this ST.

7.4.2 SAR dependencies

Table 56 SAR dependencies

Requirements	CC Dependencies	Satisfied Dependencies
ADV_ARC.1	ADV_FSP.1 and ADV_TDS.1	ADV_FSP.5, ADV_TDS.5
ADV_FSP.5	ADV_TDS.1 and ADV_IMP.1	ADV_TDS.5, ADV_IMP.2
ADV_IMP.2	ADV_TDS.3 and ALC_TAT.1 and ALC_CMC.5	ADV_TDS.5, ALC_TAT.3, ALC_CMC.5
ADV_INT.3	ADV_IMP.1 and ADV_TDS.3 and ALC_TAT.1	ADV_TDS.5, ADV_IMP.2, ALC_TAT.3
ADV_TDS.5	ADV_FSP.5	ADV_FSP.5
ADV_SPM.1	ADV_FSP.4	ADV_FSP.5
AGD_OPE.1	ADV_FSP.1	ADV_FSP.5
AGD_PRE.1	No dependencies	
ALC_CMC.5	ALC_CMS.1 and ALC_DVS.2 and ALC_LCD.1	ALC_CMS.5, ALC_DVS.2, ALC_LCD.1
ALC_CMS.5	No dependencies	
ALC_DEL.1	No dependencies	
ALC_DVS.2	No dependencies	
ALC_LCD.1	No dependencies	
ALC_TAT.3	ADV_IMP.1	ADV_IMP.2
ALC_FLR.1	No dependencies	
ASE_CCL.1	ASE_ECD.1 and ASE_INT.1 and ASE_REQ.1	ASE_ECD.1, ASE_INT.1, ASE_REQ.2
ASE_ECD.1	No dependencies	
ASE_INT.1	No dependencies	
ASE_OBJ.2	ASE_SPD.1	ASE_SPD.1
ASE_REQ.2	ASE_ECD.1 and ASE_OBJ.2	ASE_ECD.1, ASE_OBJ.2
ASE_SPD.1	No dependencies	
ASE_TSS.1	ADV_FSP.1 and ASE_INT.1 and ASE_REQ.1	ADV_FSP.5, ASE_INT.3, ASE_REQ.2
ATE_COV.3	ADV_FSP.2 and ATE_FUN.1	ADV_FSP.5, ATE_FUN.2
ATE_DPT.3	ADV_ARC.1 and ADV_TDS.4 and ATE_FUN.1	ADV_ARC.1, ADV_TDS.5, ATE_FUN.2
ATE_FUN.2	ATE_COV.1	ATE_COV.3
ATE_IND.2	ADV_FSP.2 and AGD_OPE.1 and AGD_PRE.1 and ATE_COV.1 and ATE_FUN.1	ADV_FSP.5, AGD_OPE.1, AGD_PRE.1, ATE_COV.2, ATE_FUN.2

Requirements	CC Dependencies	Satisfied Dependencies
AVA_VAN.5	ADV_ARC.1 and ADV_FSP.4 and ADV_IMP.1 and ADV_TDS.3 and AGD_OPE.1 and AGD_PRE.1 and ATE_DPT.3	ADV_ARC.1, ADV_FSP.5, ADV_IMP.2, ADV_TDS.5, AGD_OPE.1, AGD_PRE.1, ATE_DPT.1

7.5 Statement of compatibility

The statement of compatibility is described in the document [SOC]

8 TOE summary specification

8.1 SF.Firewall

The TOE implements an applet firewall according to [JCRE3], chapter 6. 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:

- 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_IFF.1/JCVM Simple security attributes
- FMT_MSA.1/JCRE Management of security attributes
- FMT_MSA.2/FIREWALL_JCVM Secure security attributes
- FMT_MSA.3/FIREWALL Static attribute initialisation
- FMT_MSA.3/JCVM Static attribute initialization
- FMT_SMF.1 Specification of Management Functions
- FMT_SMR.1 Security roles
- FDP_ROL.1/FIREWALL Basic rollback

8.2 SF.RIP

This TSF ensures that sensitive information are 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 resused by a new object creation, the previous content will be set to zero.

This TSF enforces the following SFRs:

- FDP_RIP.1/APDU 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_RIP.1/ADEL Subset residual information protection
- FDP_RIP.1/ODEL Subset residual information protection
- FDP_RIP.1/ABORT Subset residual information protection

8.3 SF.Rollback

The TOE implements atomicity and rollback mechanism for Java Card runtime environment [JCRE3] and GlobalPlatform management functions (see [GP v23]). The TOE also ensures that objects created during an aborted transaction are made unavailable.

This TSF enforces the following SFRs:

- FPT_RCV.3/Installer Automated recovery without undue loss
- FDP_ROL.1/FIREWALL Basic rollback

- FDP_ROL.1/CCM Basic rollback
- FDP_RIP.1/ABORT Subset residual information protection

8.4 SF.SCP

The TOE implements secure channel protocols according to [GP v23], chapter 10. The following protocols are supported:

- SCP02 (with options 0x15 or 0x55) according to [GP v23], appendix E
- SCP03 (with options 0x00 or 0x10) according to [GPv23 Amd D]

The SCP uses as the basic cryptographic primitives the security hardened symmetric cryptographic library which is CC EAL 6+ certified together with the underlying platform.

This TSF enforces the following SFRs:

- FDP_UIT.1/CCM Data exchange integrity
- FTP_ITC.1/SC Inter-TSF trusted channel
- FCO_NRO.2/SC Enforced proof of origin
- FDP_IFC.2/SC Complete information flow control
- FDP_IFF.1/SC Simple security attributes
- FMT_MSA.1/SC Management of security attributes
- FMT_MSA.3/SC Static attribute initialisation
- FMT_SMF.1/SC Specification of Management Functions
- FIA_UID.1/SC Timing of identification
- FIA_UAU.1/SC Timing of authentication
- FIA_UAU.4/SC Single-use authentication mechanisms
- FMT_SMR.1/SD Security roles
- FCS_COP.1 Cryptographic operation

The underlying platform supports this TSF by the following SFRs from [ST_IC].

- FCS_CKM.4/AES-SCL-1
- FCS_CKM.4/TDES-SCL-1
- FCS_COP.1/AES-SCL-1
- FCS_COP.1/TDES-SCL-1
- FCS_COP.1/CMAC-SCL-1
- FCS_CKM.4/CMAC-SCL-1

8.5 SF.CM

The TOE implements an access control policy for GlobalPlatform card management functions according to [GP v23], chapters 9.1 and 9.3 – 9.6.

This TSF enforces the following SFRs:

- FDP_ACC.1/SD Subset access control
- FDP_ACF.1/SD Security attribute based access control
- FMT_MSA.1/SD Management of security attributes
- FMT_MSA.3/SD Static attribute initialisation
- FMT_SMF.1/SD Specification of Management Functions

- FMT_SMR.1/SD Security roles
- FPT_TDC.1 Inter-TSF basic TSF data consistency
- FIA_ATD.1/AID User attribute definition
- FIA_UID.2/AID User identification before any action
- FIA_USB.1/AID User-subject binding
- FDP_ITC.2/Installer Import of user data with security attributes
- FMT_SMR.1/Installer Security roles
- FPT_RCV.3/Installer Automated recovery without undue loss
- FPT_FLS.1/Installer Failure with preservation of secure state
- 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_SMR.1/ADEL Security roles
- FPT_FLS.1/ADEL Failure with preservation of secure state

8.6 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. The core building blocks of the hardware security are the Integrity Guard and the secure coded HSL library. As an additional line of defense, the Memory Management Unit (MMU) of the hardware IC will be used for protecting the key store.

This TSF enforces the following SFRs:

- FAU_ARP.1 Security alarms
- FDP_SDI.2 Stored data integrity monitoring and action
- FPT_PHP.1 Resistance to physical attacks
- FPT_TST.1 TSF testing

The underlying platform supports this TSF by the following SFRs from [ST_IC].

- FPT_PHP.3/HSL-1
- FPT_FLS.1/HSL-1
- FDP_ITT.1
- FDP_SDC.1
- FDP_SDI.2
- FPT_FLS.1
- FPT_ITT.1
- FPT_PHP.3
- FPT_TST.1
- FRU_FLT.2
- FDP_ACC.1
- FDP_ACF.1
- FDP_MSA.1
- FDP_MSA.3
- FMT_SMF.1

8.7 SF.CS

The TOE provides cryptographic services for applets as described in 7.2.1.13 and 7.2.1.17. The cryptographic API uses as the basic cryptographic primitives the security hardened symmetric and asymmetric cryptographic library which is CC EAL 6+ certified together with the underlying platform.
The random numbers are generated by a class PTG.3 hardware number generator provided by the platform. The Java Card OS also implements especially hardened cryptographic algorithms for PACE with generic mapping and ICAO, ISO 18013 and GlobalPlatform SCP secure messaging.

This TSF enforces the following SFRs:

- FCS_CKM.1 Cryptographic key generation
- FCS_CKM.2 Cryptographic key distribution
- FCS_CKM.3 Cryptographic key access
- FCS_CKM.4 Cryptographic key destruction
- FCS_COP.1 Cryptographic operation
- FCS_COP.1/DAP Cryptographic operation
- FPR_UNO.1 Unobservability
- FCS_RNG.1

Note: FCS_CKM.1 and FCS_COP.1 comprise all CC iterations described in 7.2.1.13 and 7.2.1.17

The underlying platform supports this TSF by the following SFRs from [ST_IC].

- FCS_CKM.1/EC-1
- FCS_CKM.1/RSA-1
- FCS_COP.1/ECDH-1
- FCS_COP.1/ECDSA-1
- FCS_COP.1/RSA-1
- FCS_CKM.4/AES-SCL-1
- FCS_CKM.4/TDES-SCL-1
- FCS_COP.1/AES-SCL-1
- FCS_COP.1/TDES-SCL-1
- FCS_COP.1/CMAC-SCL-1
- FCS_CKM.4/CMAC-SCL-1
- FCS_RNG.1/HPRG

8.8 SF.PIN

The TOE implements secure PIN compare functions and integrity protection of the PIN.

- This TSF enforces the following SFRs:
- FPR_UNO.1 Unobservability
- FDP_SDI.2 Stored data integrity monitoring and action

The underlying platform supports this TSF by the following SFRs from [ST_IC].

- FDP_SDC.1
- FDP_SDI.2
- FPT_PHP.3

9 References

Number	Bibliography
[CC1]	Common Criteria Part 1, version 3.1, revision 5
[CC2]	Common Criteria Part 2, version 3.1, revision 5
[CC3]	Common Criteria Part 3, version 3.1, revision 5
[JCAPI3]	Java Card API version 3.0.5
[JCRE3]	Java Card RTE version 3.0.5
[JCVM3]	Java Card VM version 3.0.5
[PKCS v1.5]	PKCS v1.5
[PKCS v2.2]	PKCS v2.2
[JC PP]	Java Card Protection Profile Open Configuration 3.0
[JC USIM PP]	(U)SIM Java Card Platform Protection Profile 2.0.2
[SP800-67]	NIST SP 800-67
[SP800-38A]	Nist SP 800-38A
[SP800-38B]	Nist SP 800-38B
[FIPS 197]	FIPS 197-
[FIPS 180-4]	FIPS 180-4
[FIPS 186-3]	FIPS 186-3
[SEC1]	Certicom SEC1
[ISO9797]	ISO/IEC 9797
[GP v23]	GlobalPlatform Card Specification v2.3.1
[GPv23 Amd D]	GlobalPlatform v2.3.1 Amendment D
[GP-ID]	GlobalPlatform Card, ID Configuration, Version 1.0,
[DOC9303]	ICAO DOC 9303 part 11
[PP0035]	Common Criteria Protection Profile, Security IC Platform, BSI-PP-0035-2007, Version 1.0, June 2007
[PP0084]	Common Criteria Protection Profile, Security IC Platform, BSI-PP-0084-2014
[ST_IC]	Security Target for BSI-DSZ-CC-1110-V3-2020
[X9.62]	ANSI X9.62-2005
[ISO14888-3]	ISO/IEC 14888-3:2006
[ISO18013-3]	ISO/IEC 18013-3: 2009
[ISO9796-2]	ISO/IEC 9796-2:2010
[ISO9797-1]	ISO/IEC 9797-1:2011
[IEEE P1363]	IEEE Std 1363-2000
[TR 03111]	TR 03111
[RFC 5639]	RFC 5639
[AIS 31]	AIS 31, BSI
[ADMIN]	SECORA™ ID S v1.1 Administration Guide
[DBOOK]	SECORA™ ID S v1.1 Data Book
[SECGUIDE]	SECORA™ ID S v1.1 Security Guide
[SOC]	Statement of Compatibility

[SPM] Java Card firewall model in COQ

Revision History

Major changes since the last revision

Page or Reference	Description of change
1.0	First version
1.7	Release version
1.8	Change of HW certification details in section 1.4.1.1. Editorial clean-up.
1.9	Updated TOE Reference details for ‘z’ parameter in section 1.2. TOE identification description is referenced to [SECGUIDE]. Version number added to TOE name for product identification.

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2021-01-27

Published by
Infineon Technologies AG
81726 Munich, Germany

Do you have a question about this document?

Email:
dsscustomerservice@infineon.com

IMPORTANT NOTICE
The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenheitsgarantie").

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (**www.infineon.com**).

WARNINGS
Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.