Upteq NFC422 v1.0 JCS platform Security Target

Common Criteria
Security Target – Public version
EAL4+

Release	Date (dd/mm/yy)	Author	Modifications
1.2p	29/06/2020	THALES	Created from evaluated ST (V1.2)

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

1 REFERENCES

1.1 EXTERNAL REFERENCES

[CC]	Common Criteria references
[CC-1]	Common Criteria for Information Technology Security Evaluation Part 1: Introduction and general model, CCMB-2017-04-001, Version 3.1 Revision 5, April 2017.
[CC-2]	Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, CCMB-2017-04-002, Version 3.1 Revision 5, April 2017.
[CC-3]	Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, CCMB-2017-04-003, Version 3.1 Revision 5, April 2017.
[CCDB]	Common Criteria Supporting Document, Mandatory Technical Document – Composite product evaluation for Smart Cards and similar devices Ref: CCDB-2012-04-001, Version 1.5.1, May 2018.
[CEM]	Common Methodology for Information Technology Security Evaluation Methodology CCMB-2017-04-004, version 3.1 rev 5 April 2017
[JIL-SECREQ]	JIL: Security requirements for post-delivery code loading, version 1.0, February 2016
[JIL-SITE-AUDIT]	JIL: ALC Site Audit Reuse and Document Exchange Procedure version 1.3
[PP]	**Protection profiles**
[PP-IC-0084]	Security IC Platform Protection Profile with augmentation Packages– BSI-CC-PP-0084-2014
[PP-JCS-Open]	Java Card System Protection Profile – Open Configuration, Version 3.0.5 December 2017, BSI-CC-PP-0099-2017
[RGS-B1]	Référentiel Général de sécurité version 2 Annexe B1 Mécanismes cryptographiques, règles et recommandations concernant le choix et le dimensionnement des mécanismes cryptographiques; version 2.0.3 du 21 février 2014
[AIS31]	A proposal for: Functionality classes for random number generators Version 2.0 Sept 2011
[PACEPP]	Protection Profile, Machine Readable Travel Document using Standard Inspection Procedure with PACE, version 1.01, 22 July 2014. Certified and maintained by the BSI (Bundesamt für Sicherheit in der Informationstechnik) under the reference BSI-CC-PP-0068-V2-2011-MA-01.
[EAC2PP]	Protection Profile, Electronic Document implementing Extended Access Control Version 2 defined in BSI TR – 03110, Ref: BSI-CC-PP-0086, v1.01 May 20th, 2015
[PP_BAC]	"Protection Profile, Machine Readable Travel Document with “ICAO Application”, Basic Access Control, version 1.10, 25 March 2009. Certified and maintained by the BSI (Bundesamt für Sicherheit in der Informationstechnik) under the reference BSI-PP-0055-2009.
[SMG]	**Samsung References**
[ST-IC]	ST Lite of S3NSEN4/S3NSEN3 32-bit RISC Microcontroller for Smart Card including specific IC Dedicated software V1.0 20th May 2019
[CR-IC]	Certification Report ANSSI-CC-2019/29, July 31th 2019
[NIST]	**NIST references**
[FIPS180-4]	NIST, Secure Hash Standard (SHS), 2012
[FIPS PUB 186-4]	NIST, Digital Signature Standard (DSS), , 2013

[FIPS PUB 197]	Federal Information Processing Standards Publication 197 ADVANCED ENCRYPTION STANDARD (AES), 2001 November 26
[SP800-67]	NIST Special Publication 800-67 - Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher, Revision 1 – Revised January 2012.
[NIST-SP800-38A]	Recommendation for Block Cipher Modes of Operation, May 2005
[NIST-SP800-38B]	Recommendation for Block Cipher Modes of Operation, May 2005
[NIST 800-56A]	Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography, May 2013
[ISO]	**ISO references**
[ISO9796-2]	*ISO/IEC 9796-2:2010: Information technology – Security techniques – Digital Signature Schemes giving message recovery – Part 2: Integer factorization based mechanisms,* Third edition 2010-12-15
[ISO9797-1]	*ISO/IEC 9797-1:2011: Information technology – Security techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms using a block cipher,* Second edition 2011-03-01
[GP]	**Global Platform references**
[GP23]	Global Platform Card Specification Version 2.3.1 March 2018
[GP23 Amend A]	Global Platform Technology Confidential Card Content Management Card Specification v2.3 – Amendment A Version 1.2 July 2019
[GP23 Amend C]	Global Platform Technology – Contactless services – Card Specification v2.3 – Amendment C Version 1.3 July 2019
[GP23 Amend D]	Global Platform Technology Secure Channel Protocol '03' Card Specification v2.3 – Amendment D Version 1.1.2 March 2019
[GP23 Amend E]	Card Technology Security Upgrade for Card Content Management Card Specification v2.3 – Amendment E v1.1 November 2016
[GP23 Amend F]	Global Platform Technology Secure Channel Protocol '11' Card Specification v2.3 – Amendment F Version 1.2.1 – March 2019
[GP23 Amend H]	Global Platform Card Executable Load File Upgrade Card Specification v2.3 – Amendment H Version 1.0 – Feb 2017
[GP23 Privacy]	Global Platform, Privacy Framework Version 1.0 Feb 2017
[GP23 SE Config]	Global Platform, Secure Element Configuration Version 1.0 October 2012
[Others]	**Others specification references**
[TR03110-1]	Technical Guideline TR-03110-1 Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS Token – Part 1 – eMRTDs with BAC/PACEv2 and EACv1 Version 2.20, 26/02/2015
[TR03110-2]	Technical Guideline – Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS Token Part 2: Protocols for electronic Identification, Authentication and trust Services (eIDAS), v 2.21 Dec 21th 2016
[TR03110-3]	Technical Guideline – Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS Token Part 3: Common Specifications, v 2.21 Dec 21th 2016
[TR03111]	BSI: TR 03111: Elliptic Curve Cryptography, Version 2.0, 28. June 2012
[ICAO9303]	Doc 9303 Machine Readable Travel Document Part3, 2015
[PKCS1]	PKCS #1 v2.1: RSA Cryptography Standard, RSA Laboratories, June 14, 2002
[PKCS3]	RSA Laboratories: PKCS #3: Diffie-Hellman Key-Agreement Standard, An RSA Laboratories Technical Note, Version 1.4,Revised November 1, 1993

[PKI]	MRTD Technical Report, PKI for Machine Readable Travel Documents Offering ICC Read-Only Access, International Civil Aviation Organization, Version 1.1, October 01 2004
[ANSI X9.63]	ANSI-X9.63, Key Agreement and Key Transport Using Elliptic Curve Cryptography, , 2011

[JCS]	Javacard references
[JAVASPEC]	The Java Language Specification. Third Edition, May 2005. Gosling, Joy, Steele and Bracha. ISBN 0-321-24678-0.
[JVM]	The Java Virtual Machine Specification. Lindholm, Yellin. ISBN 0-201-43294-3.
[JCBV]	Java Card Platform, version 2.2 Off-Card Verifier. June 2002. White paper. Published by Sun Microsystems, Inc.
[JCRE222]	Java Card 2.2.2 Runtime Environment (JCRE) Specification – 15 March 2006 - Published by Sun Microsystems, Inc.
[JCVM222]	Java Card 2.2.2 Virtual Machine (JCVM) Specification – 15 March 2006 - Published by Sun Microsystems, Inc.
[JCAPI222]	Java Card 2.2.2 Application Programming Interface - March 2006 - Published by Sun Microsystems, Inc.
[JCRE3]	Java Card Platform, versions 3.0 up to 3.0.5, Classic Edition, Runtime Environment (Java Card RE) Specification. Published by Oracle.
[JCVM3]	Java Card Platform, versions 3.0 up to 3.0.5, Classic Edition, Virtual Machine (Java Card VM) Specification. Published by Oracle.
[JCAPI3]	Java Card Platform, versions 3.0 up to 3.0.5, Classic Edition, Application Programming Interface, Published by Oracle.
[JCRE305]	Java Card 3.0.5 Runtime Environment (JCRE) Specification, Classic Edition – May 2015 - Published by Oracle.
[JCVM305]	Java Card 3.0.5 Virtual Machine (JCVM) Specification, Classic Edition – May 2015 - Published by Oracle
[JCAPI305]	Java Card 3.0.5 Application Programming Interface, Classic Edition - May 2015 - Published by Oracle.

1.2 INTERNAL REFERENCES

[AGD]	Upteq NFC422 V1.0 Software – Guidance documentation
[AGD-PRE]	D1516186 v1.0 - Preparative guidance on CC platforms
[AGD-OPE]	Operational guidance on CC platforms With or Without CA And Optional VA, D1516184, v1.2
[AGD-OPE-VA]	D1516183 v1.0 - Operational guidance on CC platforms for VA
[Applet guidance]	Guidance for Secure application development on CC platforms, D1516182, Rev 1.1
[AGD_DAP]	Samsung_Security_Guide_DAP_Tech_Note_v1.1, July 2nd, 2019
[AGD-PATCH-DEV]	Guidance for Patch development on Thales Embedded Secure Solutions, D1341188, Rev C03
[AGD-PATCH-ADM]	Patch Loading Management for Certified Secure Elements, D1344508, Rev A01
[IDENT_CONF]	Platform Identification and Configurability UpTeq NFC422 v1.0, D1484271, Rev 1.4
[AGD-APDU]	NFC 4.2.2 v1.0_APDU Guide_D1518014A, Rev 1.0
[AGD-ARCH]	UpTeq Card Vol1 Card Architecture Guide, D1189324A, Rev 1.0
[AGD-APP-DEV]	UpTeq Card Vol4 Applet Development Guide, D1516486A, Rev 1.0
[AGD_APP-VERIF]	D1258682 C01 - Application Verification for Certified Secure Elements - External Procedure

[GP_SEC-GUIDE]	GlobalPlatform Card Composition Model Security Guidelines for Basic Applications_v2.0, GPC_GUI_050
[Others]	**Others specification references**
[ALC-DVS]	Sufficiency of security measures, R1R28368_ALC_DVS, v1.0
[ALC-DEL]	Security measures for delivery, R1R28368_ALC_DEL, v1.0

1.3 ACRONYMS AND GLOSSARY

AES	Advanced Encryption Standard
APDU	Application Protocol Data Unit
API	Application Programming Interface
CAD	Card Acceptance Device
CC	Common Criteria
CPU	Central Processing Unit
DES	Data Encryption Standard
EAL	Evaluation Assurance Level
ECC	Elliptic Curve Cryptography
EEPROM	Electrically-Erasable Programmable Read-Only Memory
ES	Embedded Software
GP	Global Platform
IC	Integrated Circuit
IT	Information Technology
JCRE	JavaCard Runtime Environment
JCS	JavaCard System
JCVM	JavaCard Virtual Machine
NVM	Non-Volatile Memory
OP	Open Platform
PIN	Personal Identification Number
PP	Protection Profile
RMI	Remote Method Invocation
RNG	Random Number Generator
ROM	Read-Only Memory
RSA	Rivest Shamir Adleman
SAR	Security Assurance Requirement
SC	Smart Card
SCP	Secure Channel Protocol
SFP	Security Function Policy
SFR	Security Functional Requirement
SHA	Secure Hash Algorithm
ST	Security Target
TOE	Target Of Evaluation
TSF	TOE Security Functionality

2 SECURITY TARGET INTRODUCTION

The main objectives of this ST are:

- To introduce TOE and the JCS Platform,
- To define the scope of the TOE and its security features,
- To describe the security environment of the TOE, including the assets to be protected and the threats to be countered by the TOE and its environment during the product development, production and usage.
- To describe the security objectives of the TOE and its environment supporting in terms of integrity and confidentiality of application data and programs and of protection of the TOE.
- To specify the security requirements which includes the TOE security functional requirements, the TOE assurance requirements and TOE security functions.

This Security Target is noted [ST-JCS] in this document

2.1 ST IDENTIFICATION

Title	Upteq NFC422 v1.0 JCS -- Security Target
Version	1.2p
ST Reference	R1R28368_JCS_ST
Author	THALES
IT Security Evaluation Facility	BRIGHTSIGHT
IT Security Certification scheme	NSCIB

2.2 TOE IDENTIFICATION

Product Name	Upteq NFC422 v1.0
Security Controllers	S3NSEN4 Rev1
TOE Name	NFC422 v1.0 JCS
TOE Version	TOE Identification Data = D0023A14C90165
TOE documentation	Guidance [AGD]
Composition elements	
Composite TOE identifier	S3NSEN4_20190520
Composite TOE Version	Rev.1

The TOE identification is available by executing a Get Data command with a proprietary Tag.
Only the part in bold, in the extract of the response below, is used to identify the Certified Product.

Tag06	T	06	Os information Tag
	L	07	Os information Length

The TOE and the product differ.

- The TOE is the JCS open platform of the Upteq NFC422 v1.0 product.
- The Upteq NFC422 v1.0 product also includes applets.

2.3 TOE OVERVIEW

2.3.1 TOE Type

The Java Card technology combines a subset of the Java programming language with a runtime environment optimized for smart cards and similar small-memory embedded devices [JCVM305]. The Java Card platform is a smart card platform enabled with Java Card technology (also called a "Java Card"). This technology allows for multiple applications to run on a single card and provides facilities for secure interoperability of applications. Applications for the Java Card platform ("Java Card applications") are called applets.

The product Upteq NFC422 v1.0 is a combo eSE/eUICC Java Card platform product addressing the consumer electronics mobile market. The both features eSE and eUICC are isolated logically (via framework) and physically (via interface/protocol).

For the present evaluation, the Target of Evaluation (TOE) is the Javacard platform part of the UpTeq NFC422 v1.0 software. The TOE boundaries encompass:

- **The Javacard System (JCS)** implemented according to the [Javacard] standard, which manages and executes applications called applets. It also provides Javacard APIs for applet development
- **The GlobalPlatform (GP) and Amendments (A, C, D, E, F, H and Privacy framework)** implemented according to the [GP] standard, which provide a common and widely used interface to communicate with a smartcard and manage applications in a secure way. It also provide GP APIs for applet development.
- **The GemActivate application**, which is the Thales proprietary solution to load and activate additional code (dedicated security on top of Global Platform one) and to activate/deactivate services post-issuance, under OEMs and Thales administration
- **The S3NSEN4 Rev1 Integrated Circuit**
- **The guidance documentation [AGD]**

2.3.2 TOE intended usage

Smart cards are used as data carriers that are secure against forgery and tampering as well as personal, highly reliable, small size devices capable of replacing paper transactions by electronic data processing. Data processing is performed by a piece of software embedded in the smart card chip, called an application.

The TOE i.e. the Java Card System is intended to transform a smart card into a platform capable of executing applications written in a subset of the Java programming language. The intended use of a Java Card platform is to provide a framework for implementing IC independent applications conceived to safely coexist and interact with other applications into a single smart card.
Applications installed on a Java Card platform can be selected for execution when the card communicates with a card reader.

Notice that these applications may contain other confidentiality (or integrity) sensitive data than usual cryptographic keys and PINs; for instance, passwords or pass-phrases are as confidential as the PIN, or the balance of an electronic purse.

So far, the most typical applications are:
– Financial applications, like Credit/Debit ones, stored value purse, or electronic commerce, among others.
– Transport and ticketing, granting pre-paid access to a transport system.
– Telephony, through the NFC chip for mobile phones.
– Personal identification, for granting access to secured sites or providing identification credentials to participants of an event.
– Loyalty programs, like the “Frequent Flyer” points awarded by airlines. Points are added and deleted from the card memory in accordance with program rules. The total value of these points may be quite high and they must be protected against improper alteration in the same way that currency value is protected.

2.3.3 NON-TOE HARDWARE/SOFTWARE/FIRMWARE REQUIRED BY THE TOE

This ST follows the Java Card PP approach, which consists in focusing on the definition of security problem, objectives and requirements that are specific to Java Card and GlobalPlatform features. Therefore, formally, non-TOE components from are the following:

- Bytecode Verifier
- In order to manage distant secure channel according to [GP23], a remote system must be able to establish a connection with TOE and therefore must possess shared secret with TOE.
- Applets are supposed to be used with the platform to communicate to external world. Applet can create a dedicated secure channel using platform services. In such case, a remote system must be able to establish a connection with applet and therefore must possess shared secret with applet.
- In order to manage local GP Privacy framework secure channel, only local terminals possessing authorization information (a shared secret stored or retrieved by terminal or secret derived from shared secret) can get access to the user data stored on the TOE and use security functionality.

2.4 TOE DESCRIPTION

2.4.1 Product Architecture

The product's design is modular. Some functionalities are mandatory features, also name "core features" and some others are considered as "plug-ins functionalities" and could be activated/deactivated/removed from the product configuration.

The high-level architecture of the UpTeq NFC422 v1.0 product can be represented as follows. In this figure, the elements in blue are configurable.

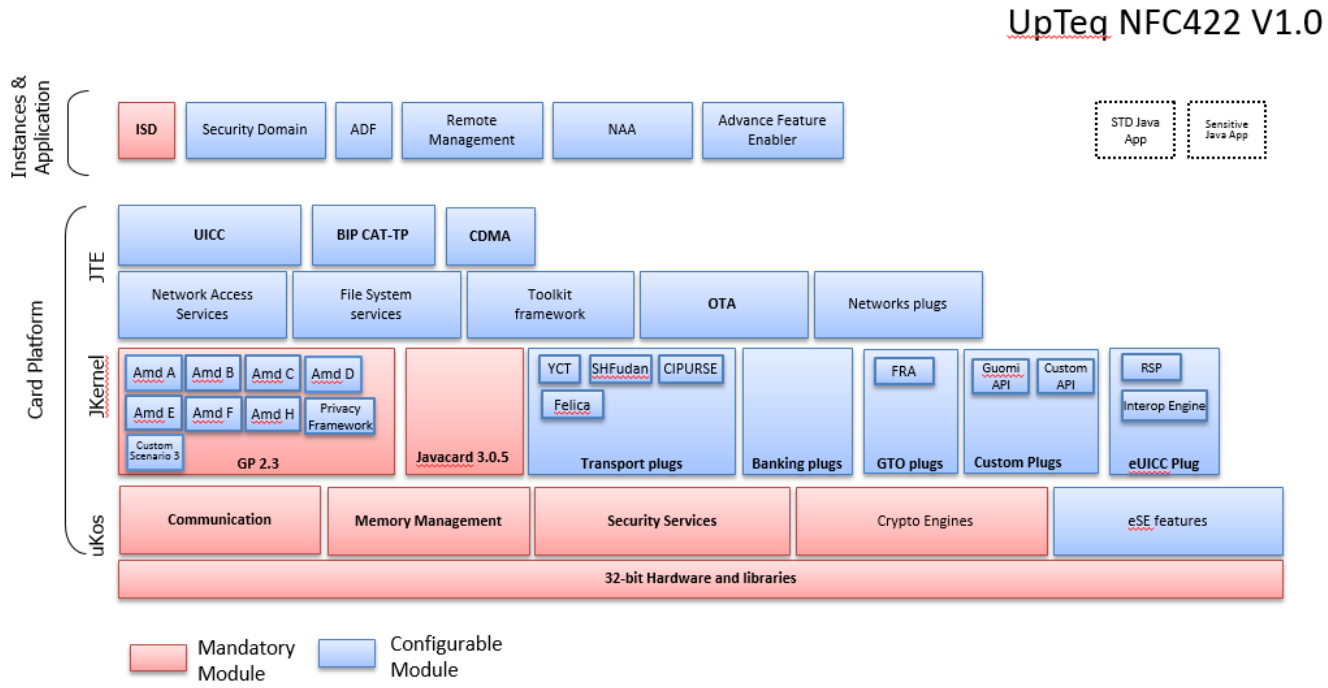


Figure 1: UpTeq NFC422 V1.0 architecture

The product's architecture can be decomposed in three layers:

IC layer

- The hardware layer composed of the S3NSEN4 Rev1 integrated circuit from Samsung LSI.

Card platform layer

- The UpTeq NFC422 v1.0 platform, which is the operating system of the product.

Application layer

- The application layer, encompassing standard and sensitive applications, as well as the security domains.

2.4.2 TOE boundaries

The following figure illustrates the evaluation boundaries for the TOE. In this figure, the TSF components have been put in red color. The other components (in blue color) do not participate to the TOE security. The two generic applets (STD Java App and Sensitive Java App) are outside of the TOE.

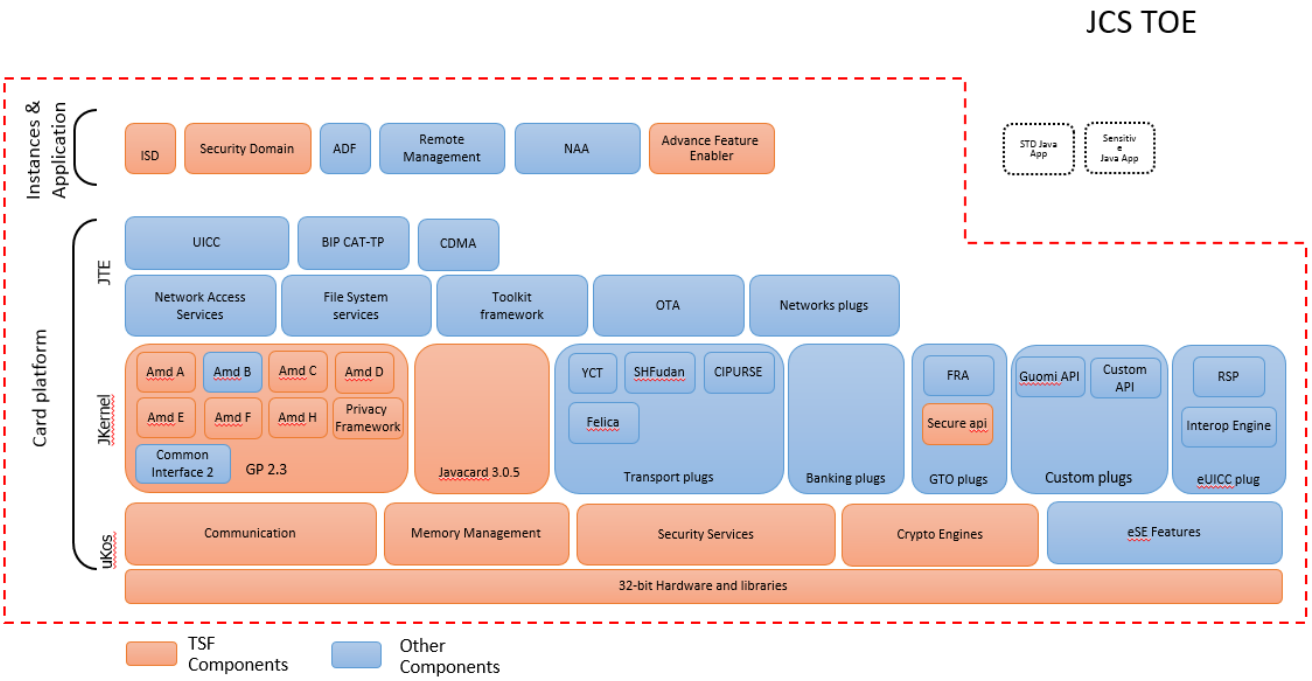


Figure 2: TOE boundaries

2.4.3 Upteq NFC422 1.0 platform description

The UpTeq NFC422 v1.0 platform is compliant with two major industry standards:

- Oracle's Java Card 3.0.5 [Javacard], which consists of the Java Card 3.0.5 Virtual Machine, Java Card 3.0.5 Runtime Environment and the Java Card 3.0.5 Application Programming Interface. Java Card RMI is not implemented in the TOE.
- Global Platform 2.3.1, Privacy Framework 1.0, Secure Element configuration 1.0 and Amendments (A, C, D, E, F, H).

This is an opened platform, meaning that additional applications can be remotely loaded and installed on the secure element "post-issuance", i.e. after the handset or mobile phone has been delivered to the end-user.
Applications can also be installed "pre-issuance" during the pre-personalization or personalization phases. Whatever the scenario (pre-issuance or post-issuance), applications' loading and installation are secured by the Global Platform security mechanisms and verification processes.

The platform implements (at least) the following services:

- Management and control of the communication between the device and external entities.
- Card basic security services as follows:
  - Checking environmental operating conditions using information provided by the IC,
  - Checking life cycle consistency,
  - Providing secure cryptography primitives and algorithms

  - Ensuring the security of the PIN and cryptographic key objects
  - Generating random numbers,
  - Handling secure data object and backup mechanisms,
  - Managing memory content,
- Enforcement of the Javacard firewall mechanism
- Garbage Collection fully implemented
- Standard Application Programming Interfaces (APIs) such as the Javacard API (JCAPI) and the Global Platform API (GPAPI)
- Proprietary Thales API: Secure API which provides security services to applications
- Initialization of the Issuer Security Domain (ISD) and management of the card life cycle
- Creation and management of Supplementary Security Domains (SSD)
- SCP02, SCP03, SCP11 support
- TDES, AES support
- RSA, ECC support
- Secure loading, installation and deletion of applications under Security Domain control
- Extradition services to allow several applications to share a dedicated Security Domain
- DAP and mandated DAP support
- Delegated Management privilege
- Trusted Path privilege
- Password Authenticated Connection Establishment (PACE)
- Extended Access Control (EAC)

The OS also permits:

- Secure loading of additional code (named OS update) with GemActivate (“Advance Feature Enabler”) application. Its activation is configurable.
- Configuration of the OS using GemActivate application

As part of a eUICC product, the UpTeq NFC422 v1.0 platform also implements a Javacard Telecom Environment (JTE). Among other features, the JTE supports one or several Network Authentication Applications (NAA), file system management, SIM toolkit, OTA and BIP functionalities, as well as UICC APIs. As shown in Figure 2, the JTE is included within the TOE but doesn’t provide any security function for the present evaluation.

2.4.4 Life-cycle

2.4.4.1 Product Life-cycle

The product life cycle is composed of the 7 phases described in Figure 3. The table below mentions the actor(s) involved in each phase.

2.4.4.1.1 Actors

The following actors are represented within the TOE.

Actors	Identification / comments
Integrated Circuit (IC) Developer	Samsung LSI
Embedded Software Developer (OS Developer) and patches Developer (if any) Applets Developer	Thales Thales or any other accredited Application Provider

Actors	Identification / comments
GemActivate Administrator	Thales, represented on the TOE by the GemActivate application and associated keys, is responsible for the remote installation of platform patches (if any) and the activation of optional platform services on the field (post-issuance).
Integrated Circuit (IC) Manufacturer	Samsung LSI
Module Manufacturer (IC packaging & testing)	Samsung LSI
Composite Product Manufacturer (Initializer/Pre-personalizer)	Samsung LSI
Personalization Agent (Personalizer)	The agent who personalizes the TOE and end-user applicative data - Samsung LSI
The Original Equipment Manufacturer (OEM) and accredited business partners (Application Providers)	The OEM, who is the issuer of the UpTeq NFC422 v1.0 product, is responsible for the secure element administration during the end-usage phase and the end of life process. The OEM also grants administration privileges to Application Providers on their respective Security Domains (APSD). Applets may be loaded onto the chip, an OS update may also be triggered at this stage.
Mobile phone holder	End-usage for mobile phone holder The end-user accesses the OEM related services and performs secure transactions with his mobile phone, thanks to the UpTeq NFC422 v1.0 hosting the sensitive applications and related assets.

Table 1: Identification of the actors

2.4.4.1.2 Life cycle description

The product life cycle phases are detailed in Figure 3. We can describe the Phases 1 to 7 as below:

- Phases 1 and 2 compose the product development: Embedded Software (IC Dedicated Software, OS, Java Card System, other platform components such as Card Manager, Applets) and IC development.
- Phase 3 and Phase 4 correspond to IC manufacturing and packaging, respectively. Some IC pre-personalization steps may occur in Phase 3.
- Phase 5 concerns the embedding of software components within the IC.
- Phase 6 is dedicated to the product personalization prior final use.
- Phase 7 is the product operational phase.

Notes related to applications

- The basic and secure applets development is part of the product life cycle, but is outside the scope of the present evaluations (since applications are out of the TOE).

- Applet loading into Flash memory can be done in phase 3, 4. Applet loading in phase 7 is also allowed. This means post-issuance loading of applets can be done for a certified JCS TOE.

Name	Phase	Actor	Description
Development	1. Embedded Software Development	Embedded Software Developer (Thales)	- Development of Java Card Platform, optional patches and applications - Generation of flash image, mapping description - Script generation for initialization and pre-personalization - Management of the TOE and pre-personalization scripts delivery process from Thales R&D to Thales PE team. Then, Thales PE provides production scripts templates to CPC team.
	2 IC Development	IC Developer (Samsung LSI)	Development of IC and associated tools
Manufacturing	3 IC Manufacturing	IC manufacturer (Samsung LSI)	Manufacturing of virgin chip integrated circuits embedding the Samsung flash Loader and protected by a dedicated transport key. JCS storage may be done at this stage.
	4 IC Packaging	Module creation (Samsung LSI)	IC packaging & testing
	5 Initialization / Pre-personalization	Composite Product manufacturer (Samsung LSI)	Product loading, based on script generated
Personalization	6 Personalization	Personalizer	Personalization and final tests
Usage	7 End-usage	Mobile phone Holder	The Consumer (Original Equipment Manufacturer) of the product is responsible for smartcard product delivery to the end-user The GemActivate Administrator, under the control of the OEM, is responsible for the remote installation of platform patches (if any) and the activation of optional platform services on the field.

Figure 3: Life cycle description

The evaluation process is limited to phases 1 to 5. The product delivery point is at the end of phase 5.

For the purpose of this evaluation, the TOE doesn't contain any patches.
For the present evaluation, the IC is manufactured at Samsung site. It is then shipped to another Samsung site where it is initialized and pre-personalized and then shipped to the Personalizer. During the shipment from Thales to Samsung, the product is protected by a diversified key.

2.4.4.2 TOE Life-cycle

The Java Card System (the TOE) life cycle is part of the product life cycle, i.e. the Java Card platform with applications, which goes from product development to its usage by the final user.

The TOE life-cycle itself can be decomposed in four stages:

- Development
- Production: Storage, pre-personalization and testing
- Preparation: Personalization and testing
- Operational Use: Final usage

Development and production of the TOE together constitute the development phase of the TOE. The development phase is subject of CC evaluation according to the assurance life cycle (ALC) class.

The TOE (JCS) storage is not necessarily a single step in the life cycle since it can be stored in parts. The JCS delivery occurs before storage and may take place more than once if the TOE is delivered in parts.
These four stages map to the product life cycle phases as shown in Figure 4.

The different guides accompanying the TOE and parts of the TOE are the ones specified in [AGD] section. They are delivered by Thales Technical representative, in form of electronic documents, using secure email with password.

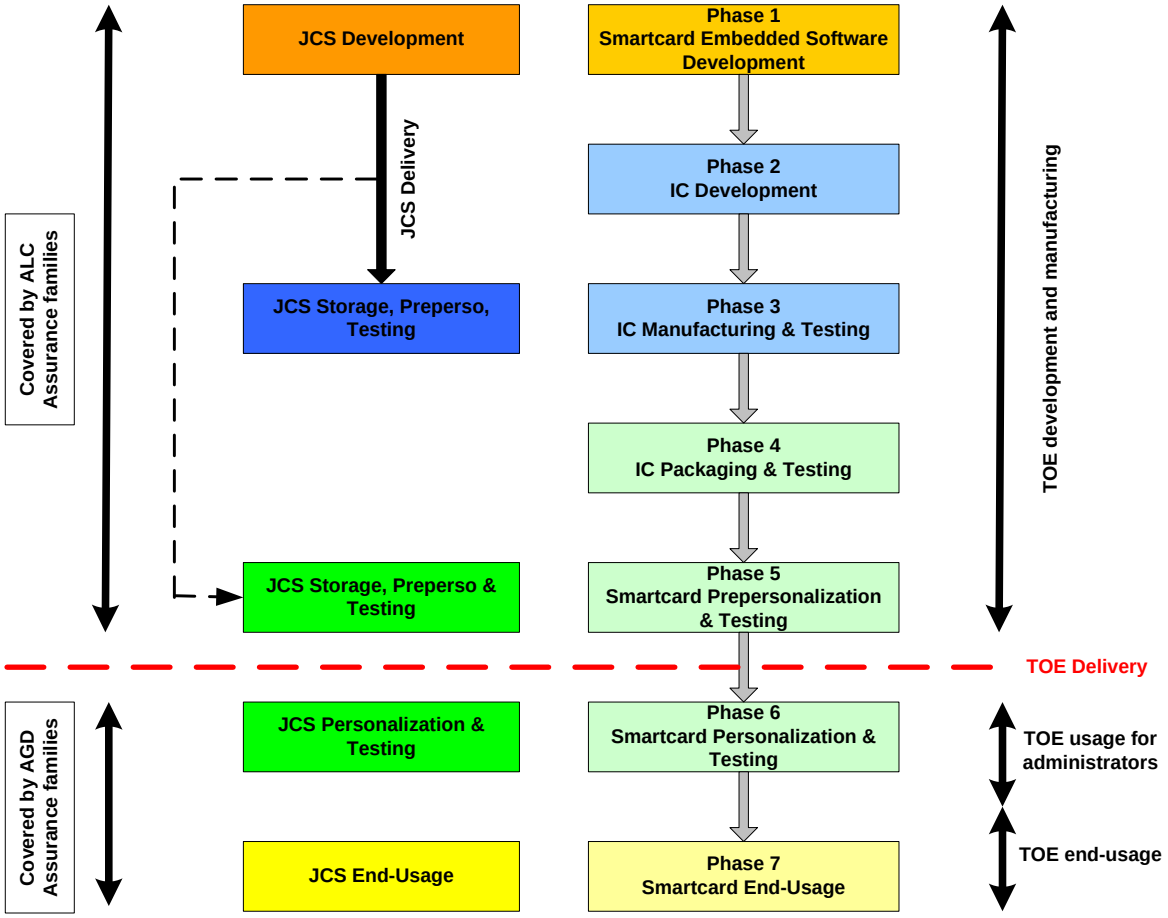


Figure 4: TOE Life Cycle within Product Life Cycle

JCS Development is performed during Phase 1. This includes JCS conception, design, implementation, testing and documentation. The JCS development shall fulfill requirements of the final product, including conformance to Java Card Specifications, and recommendations of the SCP user guidance. The JCS development shall occur in a controlled environment that avoids disclosure of source code, data and any critical documentation and that guarantees the integrity of these elements. The present evaluation includes the JCS development environment.

In Phase 3, the IC Manufacturer may store, initialize the JCS and potentially conduct tests on behalf of the JCS developer. The IC Manufacturing environment shall protect the integrity and confidentiality of the JCS and of any related material, for instance test suites. The present evaluation includes the whole IC Manufacturing environment, in particular those locations where the JCS is accessible for installation or testing. As the Security IC is certified against [PP-IC-0084] there is no need to perform the evaluation again.

In Phase 5, the SC Pre-Personalizer may store, pre-personalize the JCS and potentially conduct tests on behalf of the JCS developer. The SC Pre-Personalization environment shall protect the integrity and confidentiality of the JCS and of any related material, for instance test suites.

(Part of) JCS storage in Phase 5 implies a TOE delivery after Phase 5. Hence, the present evaluation includes the SC Pre-Personalization environment. The TOE delivery point is placed at the end of Phase 5, since the entire TOE is then built and embedded in the Security IC.

The JCS is personalized in Phase 6, if necessary. The SC Personalization environment is not included in the present evaluation. Appropriate security recommendations are provided to the SC Personalizer through the [AGD] documentation.

The JCS final usage environment is that of the product where the JCS is embedded in. It covers a wide spectrum of situations that cannot be covered by evaluations. The JCS and the product shall provide the full set of security functionalities to avoid abuse of the product by untrusted entities.

Note: Potential applications loaded in pre-issuance will be verified using dedicated evaluated verification process. Applications loaded in post-issuance will need to follow dedicated development rules.

3 CONFORMANCE CLAIMS

3.1 CC CONFORMANCE CLAIM

Common criteria Version: This ST conforms to CC Version 3.1 [CC-1] [CC-2] [CC-3]

Conformance to CC part 2 and 3:

- This ST is CC part 2 extended with the FCS_RNG, FMT_LIM.1, FMT_LIM.2 FPT_EMS.1, and FIA_API components. All the other SFRs have been drawn from the catalogue of requirements in CC part 2 [CC-2].

- This ST is CC part 3 conformant. It means that all SARs in that ST are based only upon assurance components in CC part 3 [CC-3].

Evaluation type
This is a composite evaluation, which relies on the S3NSEN4 Rev1 chip certificate and evaluation results.

S3NSEN4 chip Rev1 certificate:

- Certification done under the ANSSI scheme
- Certification report ANSSI-CC-2019/29
- Security Target [ST-IC] strictly conformant to IC Protection Profile [PP/0035]
- Common criteria version: 3.1
- Assurance level: EAL6+ (ALC_DVS.2 and AVA_VAN.5 augmentations) / ICCN0262

Consequently, the composite product evaluation (i.e. the present evaluation) includes the additional composition tasks defined in the CC supporting document “Composite product evaluation for smart cards and similar devices” [CCDB].

3.2 CONFORMANCE CLAIM TO A PACKAGE

Assurance package conformance: EAL4 augmented (EAL4+)
This ST conforms to the assurance package EAL4 augmented by ALC_DVS.2 and AVA_VAN.5.

3.3 PROTECTION PROFILE CONFORMANCE CLAIM

Protection Profile (PP) conformance claim:
This Security Target claims conformance to the [PP-JCS-Open] protection profile.

The Java Card Protection Profile makes the use of Java Card RMI and "Management of External Memory (EXT-MEM)" from the group EMG optional. The TOE does not support Java Card RMI and "Management of External Memory (EXT-MEM)" from the group EMG.

Other items relative to Smart Card Platform, Global Platform, and proprietary mechanisms (OS Update, OS Configurability, and Secure API) have been added but no conformance to additional PPs is required. Items relative to Global Privacy Framework from [EAC2PP] have been added but no conformance to [PP_EAC2] is required.

The Upteq NFC422 v1.0 JCS security target is a composite security target, including the IC security target [ST-IC]. However the security problem definition, the objectives, and the SFRs of the IC are not described in this document.

3.4 CONFORMANCE CLAIM RATIONALE

This ST conforms to [PP-JCS-Open]. The conformance is explained below:

The chapter 5.4 copies the Threats from [PP-JCS-Open]. To provide additional services to applications, extra Threats are added:

- Extra Threats associated to Global Platform, defined in chapter 5.5, are included to cover additional unauthorized card management operations (T.UNAUTHORIZED_CARD_MNGT), to cover communication channels attacks (T.COM_EXPLOIT) and to cover content management attacks (T.LIFE_CYCLE). They do not conflict with the Threats from [PP-JCS-Open].

- Extra Threats associated to additional functionalities, the PP [PP-JCS-Open] allows are optional service activation (OS configurability) and additional code loading (OS Update), defined in chapter 5.5:
  - T. UNAUTHORIZED_ACCESS_TO_ SERVICE
  - T. UNAUTHORIZED_ TOE_CODE_UPDATE
  - T.WRONG-UPDATE-STATE
  - T.CONFID-OS-UPDATE_LOAD
  - T.INTEG-OS-UPDATE_LOAD
  - T.FAKE-CERT

- Extra Threats are associated to Global Privacy Framework defined in chapter 5.6. Threats in this paragraph are refined from [PACEPP] and [EAC2PP] in a more generic form in order to be applicable to any application requiring PACE/EAC2 protocol and not only MTRD. They do not contradict the Threats from [PP-JCS-Open]:
  - T.Skimming
  - T.Eavesdropping
  - T.Abuse-Func
  - T.Information_Leakage
  - T.Phys-Tamper
  - T.Malfunction
  - T.Forgery
  - T.Sensitive_Data

All the extra Threats do not conflict with the Threats from [PP-JCS-Open].

The chapter 5.7.1 copies the OSP from [PP-JCS-Open].To provide additional services to applications, extra OSP are added:

- Extra OSP associated to Global Platform, defined in chapter 5.7.2, are included to introduce the Security domains feature of the TOE (OSP.KEY-CHANGE, OSP.SECURITY-DOMAINS, and OSP.QUOTAS). They do not conflict with the OSP from [PP-JCS-Open].

- Extra OSP associated to optional service activation (OS configurability) and additional code loading (OS Update), are defined in chapter 5.7.2. They do not conflict with the OSP from [PP-JCS-Open].
  - OSP.SERVICE_AUDIT
  - OSP.ACTIVATION_KEY_ACTOR
  - OSP.ATOMIC_ACTIVATION
  - OSP.TOE-IDENTIFICATION
  - OSP.ADDITIONAL_CODE_SIGNING
  - OSP.ADDITIONAL_CODE_ENCRYPTION

- Extra OSP introduced to manage pre-issuance are defined in chapter 5.7.2. They do not conflict with the OSP from [PP-JCS-Open].
  - OSP.TRUSTED-APPS-DEVELOPER

  - OSP.TRUSTED-APPS_PRE-ISSUANCE-LOADING

- Extra OSP related to additional services provided to applications are defined in chapter 5.7.2. They do not conflict with the OSP from [PP-JCS-Open].
  - OSP.JCAPI-Services
  - OSP.SecureAPI

- Extra OSP related to Global Privacy Framework, coming from [EAC2PP] are defined in chapter 5.7.3. They do not conflict with the OSP from [PP-JCS-Open].
  - P.Terminal
  - P.Personalisation
  - P.Manufact
  - P.Pre-Operational
  - P.EAC2_Terminal

All the extra OSP do not conflict with the OSP from [PP-JCS-Open].

The 3 assumptions from [PP-JCS-Open] are included in chapter 5.8.1. Besides these assumptions, extra assumptions are added: A.APPS-PROVIDER, A.VERIFICATION-AUTHORITY, A.CONTROLLING-AUTHORITY and A.Insp_Sys.

- The assumptions A.APPS-PROVIDER, A.VERIFICATION-AUTHORITY and A.CONTROLLING-AUTHORITY are added because Security Domains from Global Platform specification are part of the TOE.
- The assumption A.Insp_Sys is added for PACE secure channel management (Global Privacy Framework).

All the extra assumptions do not conflict with the OSP from [PP-JCS-Open].

The TOE security objectives described in chapter 6.1 represent all the security objectives for the TOE of [PP-JCS-Open]. But this ST also includes additional security objectives for the TOE.

- The Objectives listed below are Objectives for the Environment in [PP-JCS-Open]. They become Objectives for the TOE in the ST because the TOE includes the Smart Card Platform and the Card Manager. They are renamed "O.XXX" instead of "OE.XXX".
  - O.SCP.IC
  - O.SCP.RECOVERY
  - O.SCP.SUPPORT
  - O.CARD-MANAGEMENT

- The Objectives listed below are objectives for the TOE for the Secure Channel, Security Domains and Card Content Management usage, which are additional functionalities the PP [PP-JCS-Open] allows:
  - O.APPLI-AUTH
  - O.DOMAIN-RIGHTS
  - O.COMM_AUTH
  - O.COMM_INTEGRITY
  - O.COMM_CONFIDENTIALITY

- The Objectives listed below are included for the functionalities OS Update and OS Configurability, which are additional functionalities the PP [PP-JCS-Open] allows:
  - O.CONFID-OS-UPDATE.LOAD
  - O.Secure_Load_ACode
  - O.Secure_AC_Activation
  - O.TOE_Identification
  - O.REMOTE_SERVICE_ACTIVATION
  - O.REMOTE_SERVICE_AUDIT

- The Objectives listed below are relative to the APIs (Secure API proprietary and Javacard) provided to applications, which are additional functionalities the PP [PP-JCS-Open] allows:
  - O.Secure_API
  - O.JCAPI-Services

- The Objectives listed below are relative to Global Privacy Framework, which are additional functionalities the PP [PP-JCS-Open] allows. TOE objectives naming rules for this module (OT.X) is coming from [PACEPP] and remains unchanged for compatibility reason.
  - OT.AC_Pers_EAC2
  - OT.Data_Integrity
  - OT.Data_Authenticity
  - OT.Data_Confidentiality
  - OT.Identification
  - OT.Prot_Abuse_Func
  - OT.Prot_Inf_Leak
  - OT.Prot_Phys_Tamper
  - OT.Prot_Malfunction
  - OT.Sens_Data_EAC2

Moving objectives from the Environment to the TOE, adding objectives to the TOE without changing the overall objectives, do not conflict with [PP-JCS-Open].

The ST includes the Security Objectives for the Environment from [PP-JCS-Open] described in chapter 6.3.1: OE.APPLET, OE.VERIFICATION, and OE.CODE-EVIDENCE.
As already mentioned, the other Security Objectives for the Environment from [PP-JCS-Open] have been moved to Security Objectives for the TOE and renamed respectively from OE.CARD-MANAGEMENT, OE.SCP.IC, OE.SCP.RECOVERY, and OE.SCP.SUPPORT to O.CARD-MANAGEMENT, O.SCP.IC, O.SCP.RECOVERY, and O.SCP.SUPPORT.

The ST also includes additional security objectives for the Environment:

- The Objectives listed below are objectives to cover the Security Domains, Card Content Management usage, trusted actors who enable the creation, distribution and verification of applications and optionally additional code loading and activation.
  - OE.SECURITY-DOMAINS
  - OE.QUOTAS
  - OE.KEY-CHANGE
  - OE.VERIFICATION-AUTHORITY
  - OE.CONTROLLING-AUTHORITY
  - OE.APPS-PROVIDER
  - OE.TRUSTED-APPS-DEVELOPER
  - OE.TRUSTED-APPS_PRE-ISSUANCE-LOADING
  - OE.GEMACTIVATE-ADMIN

- The Objectives listed below are objectives to cover the optional OS Update functionality in order to assure that the additional code has been created by the genuine actor, not altered during the transmission phase and transmitted with confidentiality to the TOE for loading and installation. OE.Secure_ACode_Management objective defines the Key management processes related to the OS Update capability to ensure confidentiality, authenticity and integrity of the keys.
  - OE.OS-UPDATE-EVIDENCE
  - OE.OS-UPDATE-ENCRYPTION
  - OE.Secure_ACode_Management

- The Objectives listed below are objectives addressing the aspects of identified threats to be countered involving TOE's environment in Global Privacy Framework context. The objectives come from [EAC2PP]. They do not conflict with objectives from [PP-JCS-Open].
  - OE.Personalisation

- OE.Terminal
- OE.Prot_Logical_Data
- OE.User_Obligations
- OE.Chip_Auth_Key
- OE.Terminal_Authentication

All the extra Security Objectives for the Environment do not conflict with the OSP from [PP-JCS-Open].

The PP [PP-JCS-Open] focuses on the security requirements for the JCS and considers the SCP as the environment of the TOE. Nevertheless, as requested by the PP [PP-JCS-Open], this ST comprehends the IC and all the embedded software, including the OS, the JCS, as well as the additional native code and the pre-issuance applet (GemActivate application).
The chapter 7.2.1 copies all the Security Functional Requirements from [PP-JCS-Open]. Moreover, the ST adds additional threats, objectives and SFRs to fully cover and describe additional security functionality implemented in the TOE.

The extra SFRs are described in chapter 7.2.2 and linked to Smart Card Platform (chapter 7.2.2.1), card content management operations (chapter 7.2.2.2), OS Update, OS Configurability and proprietary Secure API (chapter 7.2.2.3) and Global Privacy Framework support (chapter 7.2.2.4).

The TOE restricts remote access from the CAD to the services implemented by the applets on the card to none, and as a result, the SFRs concerning Java Card RMI are not included in the ST. In the PP [PP-JCS-Open] the use of the Java Card RMI is optional. The TOE does not implement Java Card RMI.
The TOE does not permit external memory access to the services implemented by the applets on the card, and as a result the SFRs concerning "Management of External Memory" (EXT-MEM) are not included in the ST. In the PP [PP-JCS-Open], the use of the "Management of External Memory (EXT-MEM)" is optional. The TOE does not implement "Management of External Memory (EXT-MEM)".

Three Extended Components are defined compared to the PP [PP-JCS-Open]:

- Family FMT_LIM, describing the functional requirements for the Test Features of the TOE, is added to address the specific issues of preventing the abuse of functions by limiting the capabilities of the functions and by limiting their availability. This extra component does not contradict [PP-JCS-Open].
- Family FPT_EMS is added to mitigate intelligible emanations. This extra component does not contradict [PP-JCS-Open].
- Family FIA_API, describing the functional requirements for proof of the claimed identity for the authentication verification by an external entity, is added because the other families of the class FIA only address the verification of the identity of an external entity. Other families of the class FIA describe only the authentication verification of the user's identity performed by the TOE and do not describe the functionality of the TOE to prove its own identity. This extra component does not contradict [PP-JCS-Open].

The extra SFRs linked to Smart Card Platform that is, operating system and chip, the Java Card System is implemented on, are FPT_TST.1/SCP (TSF Testing), FPT_PHP.3/SCP (Resistance to physical attacks), FPT_RCV.3/SCP (Automated recovery without undue loss), FPT_RCV.4/SCP (Function recovery). The extra SFRs for SCP do not contradict the PP [PP-JCS-Open].

The extra SFRs linked to Card Content Management are added to define a policy for controlling access to card content management operations and for expressing card issuer security concerns. They add precisions for the TOE card content management. The extra SFRs for CCM are: FDP_UIT.1/CCM (to enforce the Secure Channel Protocol Information flow control policy and the Security Domain access control policy), FDP_ROL.1/CCM (to permit the rollback of the installation operation on the executable files and application instances), FDP_ITC.2/CCM (to import user data with security attributes), FPT_FLS.1/CCM (to preserve a secure state when the Security Domain fails to load/install and executable file), FCS_COP.1/DAP (to support cryptographic operation for verification of the DAP signature attached to Executable Load Applications), FDP_ACC.1/SD, FDP_ACF.1/SD,

FMT_MSA.1/SD and FMT_MSA.3/SD (to enforce the Security Domain access control policy), FMT_SMF.1/SD (to be capable of performing the Management functions as granting privileges), FMT_SMR.1/SD (to maintain the security roles: ISD, SSD, CA, VA), FTP_ITC.1/SC (to provide inter-TSF trusted channel), FCO_NRO.2/SC (to enforce proof of origin), FDP_IFC.2/SC (to complete information flow control), FDP_IFF.1/SC (to enforce the Secure Channel Protocol information flow control policy based on information security attributes like privileges and certificates), FMT_MSA.1/SC (to enforce the Secure Channel Protocol information flow control policy to restrict the ability to modify the security attributes), FMT_MSA.3/SC (to manage static attribute initialization), FMT_SMF.1/SC (to be capable of performing the management functions from Global Platform specifications), FIA_UID.1/SC and FIA_UAU.1/SC (to respectively manage Timing of identification and Timing of authentication), FIA_UAU.4/SC (to prevent reuse of authentication data related to the authentication mechanism used to open a secure communication channel with the card).
The set of SFRs that define the Card Content Management mechanism are considered to be equivalent or more restrictive than the ones from PP [PP-JCS-Open] because of the introduction of the Security Domain access control policy and Secure Channel Protocol information flow policy. These policies, by following the information flow policy defined by Global Platform specifications, provide a more restrictive implementation of the PACKAGE LOADING information flow control policy from PP [PP-JCS-Open].

The extra SFRs linked to OS Update and OS Configurability are added for proprietary platform service handled by GemActivate application. The extra SFRs in this group are: FMT_SMR.1/GemActivate (to manage security roles for GemActivate), FMT_SMF.1/GemActivate (to specify Management functions), FMT_MOF.1/GemActivate (to define security rules for Management functions behavior), FMT_MSA.1/GemActivate (to enforce the GemActivate access control SFP for management of security attributes), FMT_MTD.1/GemActivate (to specify the management of TSF data), FIA_ATD.1/OS-UPDATE (to define user attribute), FDP_ACC.1/GemActivate and FDP_ACF.1/GemActivate (to enforce the **OS Update Access Control Policy),** FMT_MSA.3/GemActivate (to manage static attribute initialization), FTP_TRP.1/OS-UPDATE (to define rules of the trusted communication path between TSF and remote users).
The set of SFRs that define the OS Update and OS Configurability mechanism are considered to be more restrictive than the ones from Card Content Management because of the introduction of the OS Update Access Control policy even more restrictive than the Security Domain access control policy and Secure Channel Protocol information flow policy defined by Global Platform specifications. And consequently, the set of SFRS linked to OS Update and OS Configurability provide a more restrictive implementation of the PACKAGE LOADING information flow control policy from PP [PP-JCS-Open].

The extra SFRs linked to Secure API are added to define security functional requirements for the proprietary Secure API. The extra SFRS in this group are FPT_FLS.1/SecureAPI (to preserve a secure state in case of failure), FPT_ITT.1/SecureAPI (to protect TSF data when transmitted between separate parts of the TOE), and FPR_UNO.1/SecureAPI (to ensure unobservability of sensitive operations by external attackers). The extra SFRs for Secure API realize additional security functionality which is allowed by the PP [PP-JCS-Open].

The extra SFRs linked to Global Privacy Framework are added for PACE, EAC1 and EAC2. The extra SFRs in this group are: FCS_CKM.1/DH_PACE (for cryptographic key generation DH for PACE and CA2 session keys), FCS_CKM.4/PACE (for cryptographic key destruction), FCS_COP.1/PACE_ENC (for Encryption / Decryption AES), FCS_COP.1/PACE_MAC (for Cryptographic operation CMAC), FCS_COP.1/PACE_CAM (for Cryptographic operation Modular Multiplication), FCS_RNG.1/PACE (for Quality metric for random numbers generation used for the authentication protocols), FIA_UAU.5/PACE, FIA_API.1/CA, FIA_UAU.5/PACE, FIA_UAU.6/CA, FIA_UAU.5/PACE, FIA_UAU.1/EAC2_Terminal and FIA_UAU.5/PACE (for authentication mechanisms used), FIA_AFL.1/PACE (to handle authentication failure), FIA_UID.1/PACE and FIA_UAU.1/PACE (for respectively Timing of identification and Timing of authentication), FIA_UAU.4/PACE (to prevent reuse of authentication data), FIA_UAU.5/PACE (to provide multiple authentication mechanisms), FIA_UAU.6/PACE (to re-authenticate the user under conditions of PACE protocol usage), FDP_RIP.1/PACE (for Inter-TSF trusted channel after PACE), FTP_ITC.1/PACE (for specification of PACE Management functions), FMT_SMF.1/PACE and FMT_SMR.1/PACE (for basic requirements to the management of the TSF data), FMT_LIM.1/PERSO, FMT_LIM.2/PERSO (for respectively limited capabilities and limited availability during personalization), FMT_MTD.1/INI_ENA, FMT_MTD.1/INI_DIS (for Management rules of TSF Initialization Data and Pre-personalization Data),

FMT_MTD.1/KEY_READ (for private key read), FPT_EMS.1, FPT_FLS.1, FPT_TST.1 and FPT_PHP.3 (to prevent inherent and forced illicit information leakage for User Data and TSF Data), FCS_COP.1/SHA and FCS_COP.1/SIG_VER (for cryptographic operations, respectively Hash for key derivation and signature verification), FIA_API.1/CA (for Authentication Proof of Identity by providing the protocol Chip Authentication 2), FIA_UID.1/EAC2_Terminal (for timing of identification), FIA_UAU.1/EAC2_Terminal (for timing of authentication), FIA_UAU.6/CA (for Re-authenticating of Terminal y the TOE), FTP_ITC.1/CA2 (for Inter-TSF trusted channel after CA2), FMT_MTD.1/Initialize_PIN (to restrict the ability to write initial PIN and PUK to personalization agent).

Regarding the structure of FCS_RNG.1/PACE SFR, even if it is related to the PACE component, the structure comes from [PP-JCS-Open]. All the SFRs attached to Global Privacy Framework, written in dedicated paragraphs, do not conflict with [PP-JCS-Open].

Extra TOE SFRs do not conflict with [PP-JCS-Open]. Such extension has no impact on PP coverage. As no other modification was done, we can conclude that the conformance is demonstrated.

4 SECURITY ASPECTS

This chapter describes the main security issues of the Java Card System and its environment addressed in this ST, 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”.

4.1 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.

Relative to OS Update:

#.CONFID-OS-UPDATE	The additional code, if any, must be kept confidential. This concerns the non-disclosure of the part of software in transit to the TOE.

4.2 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.

Relative to OS Update:

#.INTEG-OS-UPDATE	The additional code, if any, must be protected against unauthorized modification. This concerns the modification of the software patch in transit to the TOE.

4.3 UNAUTHORIZED EXECUTIONS

#.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. JCRMI being not supported by the TOE, unauthorized execution of a remote method from the CAD does not apply.
#.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, 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 neither read, write nor compare a piece of data belonging to an applet that is not in the same context, nor 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 the 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.

Relative to OS Update:

#.EXE-OS-UPDATE	The additional code, if any, must be protected against unauthorized execution. This concerns its activation before being executable.

4.4 BYTECODE VERIFICATION

#.VERIFICATION	All 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.

4.4.1 CAP file Verification

Bytecode verification includes checking at least the following properties: (3) bytecode instructions represent a legal set of instructions used on the Java Card platform; (4) adequacy of bytecode operands to bytecode semantics; (5) absence of operand stack overflow/underflow; (6) control flow confinement to the current method (that is, no control jumps to outside the method); (7) absence of illegal data conversion and reference forging; (8) enforcement of the private/public access modifiers for class and class members; (9) validity of any kind of reference used in the bytecodes (that is, any pointer to a bytecode, class, method, object, local variable, etc actually points to the beginning of piece of data of the expected kind); (10) enforcement of rules for binary compatibility (full details are given in [JCVM305], [JVM], [JCBV]). The actual set of checks performed by the verifier is implementation-dependent, but shall at least enforce all the “must clauses” imposed in [JCVM305] on the bytecodes and the correctness of the CAP files’ format.

As most of the actual Java Card VMs do not perform all the required checks at runtime, mainly because smart cards lack memory and CPU resources, CAP file verification prior to execution is mandatory. On the other hand, there is no requirement on the precise moment when the verification shall actually take place, as far as it can be ensured that the verified file is not modified thereafter. Therefore, the bytecodes can be verified either before the loading of the file on to the card or before the installation of the file in the card or before the execution, depending on the card capabilities, in order to ensure that each bytecode is valid at execution time. This Security Target assumes bytecode verification is performed off-card.

Another important aspect to be considered about bytecode verification and application downloading is, first, the assurance that every package required by the loaded applet is indeed on the card, in a binary-compatible version (binary compatibility is explained in [JCVM305]), second, that the export files used to check and link the loaded applet have the corresponding correct counterpart on the card.

4.4.2 Integrity and Authentication

Verification off-card is useless if the application package is modified afterwards. The usage of cryptographic certifications coupled with the verifier in a secure module is a simple means to prevent any attempt of modification between package verification and package installation.
Once a verification authority has verified the package, it signs it and sends it to the card. Prior to the installation of the package, the card verifies the signature of the package, which authenticates the fact that it has been successfully verified. In addition to this, a secured communication channel is used to communicate it to the card, ensuring that no modification has been performed on it.

Alternatively, the card itself may include a verifier and perform the checks prior to the effective installation of the applet or provide means for the bytecodes to be verified dynamically. On-card bytecode verifier is out of the scope of this Security Target.

4.4.3 Linking and Verification

Beyond functional issues, the installer ensures at least a property that matters for security: the loading order shall guarantee that each newly loaded package references only packages that have been already loaded on the card. The linker can ensure this property because the Java Card platform does not support dynamic downloading of classes.

4.5 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 should the installation of a package or an applet fail 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 no 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).

4.6 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 [PP-IC-0035]), 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 jeopardize the execution of the user applications. The transaction status at the beginning of an applet session must be closed (no pending updates).

Relative to OS Update:

#.SERV.OS-UPDATE	The TOE provides a means to load additional code. The GemActivate application ensures that only valid updates can be installed on the TOE.

Application note: OS Update feature is an extension of the card management defined in GP since an additional code is managed as a Java Card Package, loaded as a standard executable load file and registered with specific attributes handled with GemActivate.

Relative to OS Configurability:

#.SERV.OS-CONFIG	The TOE provides a means to activate/deactivate optional services in the JCS. The GemActivate application permits the configuration of the TOE under OEMs and Thales administration.

5 SECURITY PROBLEM DEFINITION

5.1 ASSETS FROM JAVA CARD SYSTEM PROTECTION PROFILE – OPEN CONFIGURATION

The assets of the TOE are mainly those defined in [PP-JCS-Open]. Supplementary assets have been added. The assets of [PP-IC-0084] are studied in [ST-IC].

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.

5.1.1 User data

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 and the PIN security attributes (PIN Try limit, PIN Try counter and State).
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.

5.1.2 TSF data

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.
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 monopolization and 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.

5.2 ASSETS FOR GLOBAL PLATFORM, OS UPDATE, OS CONFIGURABILITY

5.2.1 User data

D.ISD_KEYs

Issuer Security Domain cryptographic keys - Refinement of D.APP_KEYS of [PP-JCS-Open].
Used to perform card management operations on the TOE.
To be protected from unauthorized disclosure and modification.

D.APSD_KEYs

Application Provider Security Domain cryptographic keys - Refinement of D.APP_KEYS of [PP-JCS-Open].
Needed to establish secure channels with the AP. These keys can be used to load and install applications on the TOE when the Security Domain has the appropriate privileges.
To be protected from unauthorized disclosure and modification.

D.CASD_KEYs

Controlling Authority Security Domain cryptographic keys - Refinement of D.APP_KEYS of [PP-JCS-Open].
Needed to establish secure channels with the CA and to decrypt confidential content for APSDs.
To be protected from unauthorized disclosure and modification.

D.VASD_KEYs

Verification Authority Security Domain cryptographic keys - Refinement of D.APP_KEYS of [PP-JCS-Open].
Needed to verify applications Mandated DAP signature.
To be protected from unauthorized disclosure and modification.

D.CARD_MNGT_DATA

The data of the card management environment like privileges, life cycle states, memory resource quotas of security domains...
To be protected from unauthorized modification.

5.2.2 TSF data

The following assets are related to TOE identification, optional service activation and additional code management. There is no additional code associated to the present TOE, however the additional code and optional service activation mechanisms are within the evaluation scope.

D.JCS_IDENTIFIER

This is an identifier associated to the TOE used to identify it.

D.OS-UPDATE_CODE

This is the code to be added to the platform after TOE issuance. This additional code has to be signed by the OS developer. After successful verification of the signature by the Initial TOE, the additional code is loaded and installed on the TOE through an atomic activation (to create an updated TOE).
To be protected from unauthorized disclosure and modification.

D.OS-UPDATE_CODE_ID

This is an identifier associated to the additional code. The identifier is loaded in the same atomic operation than additional code loading.
To be protected from unauthorized modification.

D.OS-UPDATE_CODE_Certificate

This is a certificate which allows the TOE to check the authenticity and integrity of the additional code to be loaded.

To be protected from unauthorized modification.

D.GASD_KEYs

Thales Security Domain cryptographic keys - Refinement of D.APP_KEYS of [PP-JCS-Open].
Needed to authorize activation and OS update loading requests.
To be protected from unauthorized disclosure and modification.

D.OPTIONAL_SERVICE

Platform services can be configured by addition of optional services as e.g.

- new cryptographic algorithm service available through API
- new network authentication algorithm available through API

To be protected from unauthorized modification.

5.3 ITEMS FOR GLOBAL PRIVACY FRAMEWORK

Application note: Definition of asset associated to Global Privacy Framework is a refinement from the one in [EAC2PP] but without direct reference to travel document allowing usage of PACE and EAC secure channel for several purposes including travel document but not exclusively.
The below functions are not supported by the TOE:

- Pseudonymous Signature with all variants (PSM, PSC and PSA),
- Chip Authentication Version3 (CA3),
- Session Context,
- Static Binding,
- Certificate Extension,
- Enhance Role Authentication (ERA),
- Restricted Identification (RI).

Global Privacy Protocol SCP21 with i=01 is noted PACE in this document.
As PACE is required for EAC2, assets have been split in two groups: common assets for PACE and EAC2, and specific assets for EAC2.

5.3.1 Primary assets or user data for PACE and EAC2

Object No.	Asset	Definition	Generic security property to be maintained by the current security policy
1	user data stored on the TOE (requiring PACE secure channel)	All data (being not authentication data) being allowed to be read out solely by an authenticated terminal using PACE (in the sense of SCP 2101).	Confidentiality Integrity Authenticity
2	user data transferred between the TOE and the terminal	All data (being not authentication data) being transferred (both directions) between the TOE and an authenticated terminal using PACE (in the sense of (SCP 2101).	Confidentiality Integrity Authenticity

Table 2: Primary Assets

Note: As PACE is required for EAC2, PACE assets are also EAC2 assets.

5.3.2 Primary assets or user data for EAC2

Object No.	Asset	Definition	Generic security property to be

			maintained by the current security policy
5	Sensitive user data	All data (being not authentication data) being allowed to be accessed by an EAC2 terminal with appropriate authorization level. Note: Sensitive user data are a subset of all user data protected by PACE and EAC2 and stored in TOE.	Confidentiality Integrity Authenticity
6	Sensitive User Data stored on the TOE (EAC2)	All data with the exception of authentication data (as electronic document but not limited to), that are stored in the context of the application(s) using EAC2. These data are allowed to be accessed by an EAC2 terminal with appropriate authorization level. Note: This asset is an extension of the asset defined in [PACEPP].	Confidentiality Integrity Authenticity
7	Sensitive User Data transferred between the TOE and the Terminal – (EAC2)	All data with the exception of authentication data (as electronic document but not limited to), that are transferred (both directions) during usage of the application(s) using EAC2 between the TOE and authenticated EAC2 terminals. Note: This asset is an extension of the asset defined in [PACEPP].	Confidentiality Integrity Authenticity

Table 3: Primary Assets

Note: Unavailability in a sense of non-disclosure of data allowing user traceability.

5.3.3 Secondary assets and TSF data for PACE and EAC2

The secondary assets also having to be protected by the TOE in order to achieve a sufficient protection of the primary assets are listed in the following table. The secondary assets represent TSF and TSF-data in the sense of the CC.

Object No.	Asset	Definition	Generic security property to be maintained by the current security policy
3	Accessibility to the PACE TOE functions and data only for authorised subjects by PACE	Property of the TOE to restrict access to TSF and TSF-data stored in the TOE to authorised subjects only by PACE.	Availability
4	PACE establishment authorization data	Restricted-revealable authorization information for a human user being used for verification of the authorization attempts as authorized user (PACE password: MRZ, CAN, PIN, PUK). These data are stored in the TOE and are not to be send to it.	Confidentiality Integrity

5	TOE internal PACE secret cryptographic keys	Permanently or temporarily stored secret cryptographic material used by the TOE in order to enforce PACE security functionality.	Confidentiality Integrity
6	TOE internal PACE non-secret cryptographic material	Permanently or temporarily stored non-secret cryptographic (public) keys and other non-secret material (Document Security Object SOD containing digital signature) used by the TOE in order to enforce PACE security functionality.	Integrity Authenticity

Table 4: Secondary Assets for PACE

Note: PACE passwords are not to be sent to the TOE.

5.3.4 Secondary assets and TSF data for EAC2

The secondary assets also having to be protected by the EAC2 features of TOE in order to achieve a sufficient protection of the EAC2 primary assets are listed in the following table. The secondary assets represent TSF and TSF-data in the sense of the CC.

Object No.	Asset	Definition	Generic security property to be maintained by the current security policy
3b	Accessibility to the TOE functions and data only for authorised subjects by EAC2	Property of the TOE to restrict access to TSF and TSF-data stored in the TOE to authorised subjects only by EAC2.	Availability
4b	EAC2 establishment authorization data	Restricted-revealable authorization information for a human user being used for verification of the authorization attempts as authorized user (EAC2). These data are stored in the TOE and are not to be send to it.	Confidentiality Integrity
5b	TOE internal secret EAC2 cryptographic keys	Permanently or temporarily stored secret cryptographic material used by the TOE in order to enforce the EAC2 security functionality.	Confidentiality Integrity
6b	TOE internal EAC2 non-secret cryptographic material	Permanently or temporarily stored non-secret cryptographic (public) keys and other non-secret material (Document Security Object SOD containing digital signature) used by the TOE in order to enforce EAC2 security functionality.	Integrity Authenticity

Table 5: Secondary Assets for EAC2

Note: PACE passwords are not to be sent to the TOE.

5.3.5 Subjects and external entities

The ST considers the following external entities and subjects for PACE and EAC2 usage:

External Entity No.	Role	Definition

1	Application user (e.g. travel document holder)	This entity is commensurate with application user for whom the Issuer has personalised the PACE part of the TOE and therefore may use PACE secure channel (e.g. 'Electronic Document Holder' in [EAC2PP]
2	Application user (e.g. travel document presenter)	This entity is commensurate with application user with usage of PACE secure channel to be authenticated (e.g. 'Electronic Document Presenter' in [EAC2PP]
3	Terminal	A terminal is any technical system communicating with the TOE through the contactless/contact interface and being recognised by the TOE as not being PACE or EAC2 authenticated.
4	PACE Terminal	A local system communicating with the TOE and implementing the terminal's part of the PACE protocol. This entity is commensurate with PACE Terminal in [EAC2PP].
5	EAC2 Terminal	A terminal that has successfully passed Terminal Authentication 2 is an EAC2 terminal. It is authorized to access a subset or all of the data stored on the electronic document.
6	Personalisation Agent	This entity is commensurate with 'Personalisation agent' in [PACEPP] and in [EAC2PP].
7	Manufacturer	This entity is commensurate with 'IC Manufacturer' and FF Manufacturer and Pre-personalizer roles as defined in §2.5.1.2 Life cycle description. This entity is commensurate with 'Manufacturer' in [PACEPP] and in [EAC2PP].
8	Attacker	This external entity is commensurate with 'Attacker' in [EAC2PP].

Table 6: Subjects and External Entities

5.4 THREATS FROM JAVA CARD SYSTEM PROTECTION PROFILE – OPEN CONFIGURATION

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.

5.4.1 Confidentiality

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**.

5.4.2 Integrity

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_KEYs**.

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.

5.4.3 Identity usurpation

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

5.4.4 Unauthorized execution

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 security function such as the firewall. See #.NATIVE for details.
Directly threatened asset(s): **D.JCS_DATA**.

5.4.5 Denial of Service

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**.

5.4.6 Card management

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 (p 343) 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).

5.4.7 Services

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**.

5.4.8 Miscellaneous

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.

5.5 THREATS ASSOCIATED TO GLOBAL PLATFORM, OS UPDATE, OS CONFIGURABILITY

The following threats are related to Global platform.

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.ISD_KEYS, D.CASD_KEYS, D.VASD_KEYS, D.APSD_KEYS, D.APP_C_DATA, D.APP_I_DATA, D.APP_CODE** and **D.CARD_MNGT_DATA.**

T.COM_EXPLOIT

An attacker remotely exploits the communication channels established between the TOE and a third party in order to modify or disclose confidential data.
All assets are threatened.

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 re-personalizes the application).
Directly threatened asset(s): **D.APP_I_DATA, D.APP_C_DATA** and **D.CARD_MNGT_DATA**.

The following threats are related to optional service activation (OS configurability) and additional code loading (OS Update).

T. UNAUTHORIZED_ACCESS_TO_ SERVICE

The attacker may gain direct access to an optional platform service without authorization by bypassing access control to service activation.

Directly threatened asset(s): **D.GASD_KEYS**, **D.OPTIONAL_SERVICE**.

T. UNAUTHORIZED_ TOE_CODE_UPDATE

The attacker attempts to load malicious additional code in order to compromise the security features of the TOE.

Directly threatened asset(s): **D.JCS_CODE, D.JCS_DATA,** and **D.OS-UPDATE_CODE.**

T.WRONG-UPDATE-STATE

The attacker prevents the OS update operation to be performed atomically, resulting in an inconsistency between the resulting TOE code and the identification data:

- The additional code is not loaded within the TOE, but the identification data is updated to mention that the additional code is present
- The additional code is loaded within the TOE, but the identification data is not updated to indicate the change.

Directly threatened asset(s): **D.JCS_IDENTIFIER**

T.CONFID-OS-UPDATE_LOAD

The attacker discloses (part of) the additional code to be used to update the TOE when it is transmitted to the TOE for installation. See #.CONFID-OS-UPDATE for details.

Directly threatened asset(s): **D.JCS_CODE, D.JCS_DATA,** and **D.OS-UPDATE_CODE**.

T.INTEG-OS-UPDATE_LOAD

The attacker modifies (part of) the additional code to be used to update the TOE when it is transmitted to the TOE for installation. See #.INTEG-OS-UPDATE for details.

Directly threatened asset(s): **D.JCS_CODE, D.JCS_DATA,** and **D.OS-UPDATE_CODE**.

T.FAKE-CERT

The attacker modifies the certificate used by the TOE to verify the signature of the additional code. Hence he is able to sign and successfully load malicious additional code inside the TOE.

Directly threatened asset(s): **D.JCS_CODE, D.JCS_DATA, D.OS-UPDATE_CODE_Certificate** and **D.OS-UPDATE_CODE**.

5.6 THREATS ASSOCIATED TO GLOBAL PRIVACY FRAMEWORK

Application note: Threats in this paragraph are refined form [PACEPP] and [EAC2PP] in a more generic form in order to be applicable to any application requiring PACE/EAC2 protocol and not only MTRD.

5.6.1 Threats related to PACE AND EAC2

These threats are associated to Common assets to PACE and EAC2.

T.Skimming Capturing Card-Terminal Communication

Adverse action: An attacker imitates a PACE terminal (e.g. inspection system) in order to get access to the user data stored on or transferred between the TOE and the use (e.g. inspecting authority) connected via the contactless/contact interface of the TOE.

Threat agent: having high attack potential, cannot read and does not know the correct value of the shared password (PACE password) in advance.

Asset: confidentiality of application data (e.g. logical travel document data).

Application Note 11: MRZ is printed and CAN is printed or stuck on the travel document.
Please note that neither CAN nor MRZ effectively represent secrets, but are restricted-revealable, cf. OE.User_Obligations.

T.Eavesdropping Eavesdropping on the communication between the TOE and the PACE terminal

Adverse action: An attacker is listening to the communication between the TOE (e.g. travel document) and the PACE authenticated terminal (e.g. BIS-PACE) in order to gain the user data transferred between the TOE and the terminal connected.

Threat agent: having high attack potential, cannot read and does not know the correct value of the shared password (PACE password) in advance.
Asset: confidentiality of application data (e.g. logical travel document data).

T.Abuse-Func Abuse of Functionality

Adverse action: An attacker may use functions of the TOE which shall not be used in TOE operational phase in order (i) to manipulate or to disclose the User Data stored in the TOE, (ii) to manipulate or to disclose the TSF-data stored in the TOE or (iii) to manipulate (bypass, deactivate or modify) soft-coded security functionality of the TOE. This threat addresses the misuse of the functions for the initialization and personalization in the operational phase after delivery to the Application user*.

Threat agent: having high attack potential, being in possession of one or more legitimate application data requiring PACE usage (e.g. travel document for MRTD).

Asset: integrity and authenticity of the application data requiring PACE usage (e.g. travel document for MRTD), availability of the functionality for the application data requiring PACE usage (e.g. travel document for MRTD).

Application note: for MRTD, Application user* is travel document holder

T.Information_Leakage Information Leakage from travel document

Adverse action: An attacker may exploit information leaking from the TOE during its usage in order to disclose confidential User Data or/and TSF-data stored on the TOE and associated applications (e.g. travel document) or/and exchanged between the TOE and the terminal connected. The information leakage may be inherent in the normal operation or caused by the attacker.

Threat agent: having high attack potential

Asset: confidentiality of User Data and TSF-data including associated applications data requiring PACE usage (e.g. travel document for MRTD).

T.Phys-Tamper Physical Tampering

Adverse action: An attacker may perform physical probing of the TOE and associated applications (e.g. travel document) in order (i) to disclose the TSF-data, or (ii) to disclose/reconstruct the TOE's Embedded Software. An attacker may physically modify the TOE and associated applications (e.g. travel document) in order to alter (I) its security functionality (hardware and software part, as well), (ii) the User Data or the TSF-data stored on the TOE and associated application data (e.g. travel document).

Threat agent: high attack potential, being in possession of one or more legitimate TOE and associated applications (e.g. travel documents).

Asset: integrity and authenticity of the TOE and associated application data (e.g. travel document), availability of the functionality of the TOE and associated application data (e.g. travel document), confidentiality of User Data and TSF-data of the TOE and associated application data (e.g. travel document)

T.Malfunction Malfunction due to Environmental Stress

Adverse action: An attacker may cause a malfunction of the TOE (hardware and software) and associated applications by applying environmental stress in order to (i) deactivate or modify security features or functionality of the TOE' hardware or to (ii) circumvent, deactivate or modify security functions of the TOE's Embedded Software. This may be achieved e.g. by operating the TOE and associated applications (e.g. travel document) outside the normal operating conditions, exploiting errors in the TOE and associated applications (e.g. travel document) Embedded Software or misusing administrative functions. To exploit these vulnerabilities an attacker needs information about the functional operation.

Threat agent: having high attack potential, being in possession of one or more legitimate TOE and associated applications (e.g. travel documents), having information about the functional operation

Asset: integrity and authenticity of the TOE and associated applications (e.g. travel document), availability of the functionality of the TOE and associated applications (e.g. travel document), confidentiality of User Data and TSF-data of the TOE and associated applications (e.g. travel document).

Application note: A malfunction of the TOE may also be caused using a direct interaction with elements on the chip surface. This is considered as being a manipulation (refer to the threat T.Phys-Tamper) assuming a detailed knowledge about TOE's internals.

T.Forgery Forgery of Data

Adverse action: An attacker fraudulently alters the User Data or/and TSF-data stored on TOE or associated application (e.g. the travel document) or/and exchanged between the TOE and the terminal connected in order to outsmart the PACE authenticated terminal (e.g. BIS-PACE by means of changed Application user data*.The attacker does it in such a way that the terminal connected perceives these modified data as authentic one.

Threat agent: having high attack potential

Asset: Integrity of the travel document

Application note: Application user data is travel document holder data for MRTD (e.g. biographic or biometric data)

5.6.2 Threats related to EAC2

The threats in this chapter are associated to EAC2 only.

T.Sensitive_Data Unauthorized access to sensitive user data
Adverse action: An attacker tries to gain access to sensitive user data through the communication interface of the electronic document's chip.
The attack T.Sensitive_Data is similar to the threat T.Skimming from [PACEPP] w.r.t. the attack path (communication interface) and the motivation (to get data stored on the electronic document's chip) but differs from those in the asset under the attack (sensitive data vs. digital MRZ, digitized portrait and other data), the opportunity (i.e. knowing the PACE Password) and therefore the possible attack methods.
Threat agent: having high attack potential, knowing the PACE Password, being in possession of a legitimate electronic document
Asset: confidentiality of sensitive user data stored on the electronic document

5.7 ORGANIZATIONAL SECURITY POLICIES

5.7.1 OSP from Java Card System Protection Profile – Open Configuration

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

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 recommendations related to the isolation property of the platform, this policy shall also ensure that the verification authority checks that these recommendations are applied in the application code.

5.7.2 OSP associated to Global Platform, OS Update, OS Configurability

The following OSPs shall also be considered for the present evaluation.

- OSP.KEY-CHANGE, OSP.SECURITY-DOMAINS and OSP.QUOTAS are related to Global platform services management
- OSP.SERVICE_AUDIT and OSP.ACTIVATION-KEY-ACTOR are provided to manage optional service activation.
- OSP.ATOMIC_ACTIVATION, OSP.TOE_IDENTIFICATION, OSP.ADDITIONAL_CODE_SIGNING and OSP.ADDITIONAL_CODE_ENCRYPTION are provided to manage additional code loading and activation.

- OSP.TRUSTED-APPS-DEVELOPER and OSP.TRUSTED-APPS_PRE-ISSUANCE-LOADING are provided to manage pre-issuance.
- OSP.SecureAPI and OSP.JCAPI-Services are related to additional services provided to applications by the TOE.

OSP.KEY-CHANGE

The AP shall change its initial security domain keys (APSD) before any operation on its Security Domain.

OSP. SECURITY-DOMAINS

Security domains can be dynamically created, deleted and blocked during usage phase in post-issuance mode.

OSP. QUOTAS

Security domains are subject to quotas of memory at creation.

OSP.SERVICE_AUDIT

The GemActivate administrator (Thales or OEM) can audit optional platform service activation using remote service audit.

OSP.ACTIVATION_KEY_ACTOR

The key actor is a trusted actor in charge of the secure storage of the activation keys generated and stored outside of the TOE and imported in the TOE by the TOE personalizer during initial personalization. He ensures the security of the keys for remote service activation.

OSP.ATOMIC_ACTIVATION

As the TOE supports ability to include additional code, additional code has to be loaded and installed on the initial TOE through an atomic activation to create the Updated TOE.

Each additional code shall be identified with unique Identification Data. During such atomic activation, identification Data of the TOE have to be updated to clearly identify the Updated TOE.

In case of interruption or incident during activation, the TOE shall remain in its initial state or fail secure.

OSP.TOE-IDENTIFICATION

Identification Data of the resulting updated TOE shall identify the Initial TOE and the activated additional code. Identification Data shall be protected in integrity.

OSP.ADDITIONAL_CODE_SIGNING

The additional code has to be signed with a cryptographic key according to relevant standard and the generated signature is associated to the additional code.

The additional code signature must be checked during loading to assure its authenticity and integrity and to assure that loading is authorized on the TOE.

The cryptographic key shall be of sufficient quality and the process of key generation and certificate generation shall be appropriately secured to ensure (i) the confidentiality, authenticity and integrity of the key, (ii) the authenticity and integrity of certificate.

OSP.ADDITIONAL_CODE_ENCRYPTION

The additional code has to be encrypted according to relevant standard in order to ensure its confidentiality when it is transmitted to the TOE for loading and installation.

The encryption key shall be of sufficient quality and the process of key generation shall be appropriately secured to ensure the confidentiality, authenticity and integrity of the key.

OSP.TRUSTED-APPS-DEVELOPER

There are application developers (as Thales) considered as trusted by Application Providers. The confidence in these actors has been obtained by audits of the development process and development environment performed by ITSEF during private scheme evaluations or Common Criteria composite evaluations.

As a consequence, the development process applied by a trusted developer provides confidence that applications developed by this actor are not aggressive versus the platform and other applications loaded on top of it.

OSP.TRUSTED-APPS_PRE-ISSUANCE-LOADING

For Pre-Issuance loading of trusted* applications, the process audited by ITSEF during private scheme evaluations or Common Criteria evaluations must be used.

* Application notes:

- An application is considered as trusted if it has been developed or verified by a trusted actor (as Thales).
- An application developed by a third party, not managed by a trusted actor, can be considered as a trusted application only if it has been signed and verified by SD through DAP privilege.

* Note:

- For non-trusted actors, any SD intending to receive a package (whether created pre-issuance or post issuance) must be created with DAP privilege and DAP key.

OSP.JCAPI-Services

This OSP is enforced by the TOE security objective O.JCAPI-Services.

OSP.SecureAPI

The TOE must contribute to ensure that applications can optimize control on their sensitive operations. For that purpose, the TOE implements a dedicated API which provides security services to applications (e.g. secure array management, loss of data integrity detection, inconsistent execution flow detection, reaction against tearing or fault induction).

5.7.3 OSP associated to Global Privacy Framework

The TOE shall comply with the following Organizational Security Policies (OSPs) as security rules, procedures, practices, or guidelines imposed by an organization upon its operations.
Note: OSP naming rules for this module (P.X) is coming from [EAC2PP] and remains unchanged for compatibility reason.

5.7.3.1 OSP for PACE AND EAC2

P.Terminal Abilities and trustworthiness of terminals

The Basic Inspection Systems with PACE shall operate their terminals as follows:

1.) The related terminals (basic inspection system, cf. above) shall be used by terminal operators and by Applicative users as defined in [PKI].

2.) They shall implement the terminal parts of the PACE protocol, of the Passive Authentication [PKI] and use them in this order. The PACE terminal shall use randomly and (almost) uniformly selected nonces, if required by the protocols (for generating ephemeral keys for Diffie-Hellmann).

3.) The related terminals need not to use any own credentials.

4.) The related terminals and their environment shall ensure confidentiality and integrity of respective data handled by them (e.g. confidentiality of PACE passwords, etc.), where it is necessary for a secure operation of the TOE.

Application note: Applicative user is travel document holder in MTRD context.

P.Personalisation Personalisation of the applicative data by authorized issuing actor only
The issuer* guarantees the correctness of the user data to be included in TOE in Personalisation phase. In particular, the issuer* guarantees user data are consistent with respect of the end user of the TOE.

Application note: For MRTD application, the issuer is here “issuing State or Organisation”, the user data includes at least, “the biographical data, the printed portrait and the digitized portrait, the biometric reference data and other data of the logical travel document” and the end user is “the travel document holder”. The personalisation of the travel document for the holder is performed by an agent authorized by the issuing State or Organisation only.

P.Manufact Manufacturing of the TOE with Initialization Data for application.

The Initialization Data are written by the IC Manufacturer to identify the IC uniquely. The FF Manufacturer writes the Pre-personalisation Data which contains at least the Personalisation Agent Key.

P.Pre-Operational Pre-operational handling of the TOE and associated applications

1.) The Issuer issues the TOE and associated applications (e.g. travel document) and approves it using the terminals complying with all applicable laws and regulations.

2.) The Issuer guarantees correctness of the user data (amongst other of those, concerning the application user (e.g.travel document holder) and of the TSF-data permanently stored in the TOE[1].

3.) The Issuer uses only such TOE’s technical components (IC) which enable traceability of the TOE and associated applications (e.g. travel documents) in their manufacturing and issuing life cycle phases, i.e. before they are in the operational phase.

If the Issuer authorises a Personalisation Agent to personalise the TOE and associated applications (e.g. travel documents) for application user (e.g. travel document holder), the Issuer has to ensure that the Personalisation Agent acts in accordance with the Issuer’s policy.

5.7.3.2 OSP for EAC2

The TOE shall comply with the following Organizational Security Policies (OSPs) as security rules, procedures, practices, or guidelines imposed by an organization upon its operations.

P.EAC2_Terminal Abilities of Terminals executing EAC Version 2

Terminals that intent to be EAC2 terminals must implement the respective terminal part of the protocols required to execute EAC version 2 according to [TR03110-2], and store (static keys) or generate (temporary keys and nonces) the corresponding credentials.

P.Restricted Identity (Restricted Identity and Sector’s Static Key Pairs) is not applicable because the TOE does not support the Restricted Identity protocol.

5.8 ASSUMPTIONS

5.8.1 Assumptions from Java Card System Protection Profile – Open Configuration

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

A.APPLET

Applets loaded post-issuance do not contain native methods. The Java Card specification explicitly "does not include support for native methods" ([JCVM222], §3.3) outside the API.

A.DELETION

Deletion of applets through the card manager is secure. Refer to #.DELETION for details on this assumption.

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.

5.8.2 Assumptions associated to Global Platform, OS Update, OS Configurability

The following assumptions shall also be considered for the present evaluation.

A.APPS-PROVIDER

The AP is a trusted actor that provides basic or secure applications. He is responsible for his security domain keys (APSD keys).
Application note: An AP generally refers to the entity that issues the application. For instance it can be a financial institution for a payment application such as EMV or a transport operator for a transport application such as Calypso.

A.VERIFICATION-AUTHORITY

The VA is a trusted actor who is able to guarantee and check the digital signature attached to a basic or secure application.
Application note: As a consequence, it guarantees the success of the application validation or certification upon loading.

A.CONTROLLING-AUTHORITY

The CA is a trusted actor responsible for securing the APSD keys creation and personalization. He is responsible for his security domain keys (CASD keys).

5.8.3 Assumptions associated to Global Privacy Framework

A.Insp_Sys Inspection Systems for global interoperability

The Extended Inspection System (EIS) for global interoperability implements at least the terminal part of PACE. If several protocols are supported by the EIS, PACE secure channel must be established and applicative data (e.g. the logical travel document) must be transferred under PACE. Other operations may be done when additional protocols are supported by the terminal.

6 SECURITY OBJECTIVES

6.1 SECURITY OBJECTIVES FOR THE TOE FROM JAVA CARD SYSTEM PROTECTION PROFILE – OPEN CONFIGURATION

This section defines the security objectives to be achieved by the TOE.

6.1.1 Identification

O.SID
The TOE shall uniquely identify every subject (applet, or package) before granting it access to any service.

6.1.2 Execution

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.

6.1.3 Services

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.RNG
The TOE shall ensure the cryptographic quality of random number generation. For instance random numbers shall not be predictable and shall have sufficient entropy.

The TOE shall ensure that no information about the produced random numbers is available to an attacker since they might be used for instance to generate cryptographic keys.

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 (including the PIN try limit, PIN try counter and states). If the PIN try limit is reached, no further PIN authentication must be allowed.
See #.PIN-MNGT for details.
Application 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 limit and the try counter's value are as sensitive as that of the PIN and the TOE must restrict their modification only to authorized applications such as the card manager.

O.TRANSACTION
The TOE must provide a means to execute a set of operations atomically. See #.TRANSACTION for details.

O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION, O.RNG and O.CIPHER are actually provided to applets in the form of Java Card APIs. Vendor-specific libraries can also be present on the card and made available to applets; those may be built on top of the Java Card API or independently. These proprietary libraries will be evaluated together with the TOE.

6.1.4 Object deletion

O.OBJ-DELETION
The TOE shall ensure the object deletion shall not break references to objects. See #.OBJ-DELETION for further details.

6.1.5 Applet management

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.

Application 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 codes 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.

6.2 ADDITIONAL SECURITY OBJECTIVES FOR THE TOE

This section defines the additional security objectives to be achieved by the TOE.

6.2.1 SCP

The Objectives described in this section are Objectives for the Environment in [PP-JCS-Open]. They become Objectives for the TOE because the TOE in this ST includes the SCP.

O.SCP.IC
The SCP shall provide all IC security features against physical attacks.
This security objective for of the TOE refers to the point (7) of the security aspect #.SCP:

- It is required that the IC is designed in accordance with a well-defined set of policies and Standards (likely specified in another protection profile), and will be tamper resistant to actually prevent an attacker from extracting or altering security data (like cryptographic keys) by using commonly employed techniques (physical probing and sophisticated analysis of the chip). This especially matters to the management (storage and operation) of cryptographic keys.

O.SCP.RECOVERY
If there is a loss of power, or if the smart card is withdrawn from the CAD while an operation is in progress, the SCP must allow the TOE to eventually complete the interrupted operation successfully, or recover to a consistent and secure state.
This security objective of the TOE 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 of the TOE refers to the security aspects 2, 3, 4 and 5 of #.SCP:
(2) It does not allow the TSFs to be bypassed or altered and does not allow access to other low-level functions than those made available by the packages of the API. That includes the protection of its private data and code (against disclosure or modification) from the Java Card System.
(3) It provides secure low-level cryptographic processing to the Java Card System.
(4) It supports the needs for any update to a single persistent object or class field to be atomic, and possibly a low-level transaction mechanism.
(5) It allows the Java Card System to store data in "persistent technology memory" or in volatile memory, depending on its needs (for instance, transient objects must not be stored in non-volatile memory). The memory model is structured and allows for low-level control accesses (segmentation fault detection).

6.2.2 CMGR

The Objectives described in this section are Objectives relative to the card management.
O.CARD-MANAGEMENT is an objective for the Environment in [PP-JCS-Open]. It becomes Objective for the TOE because the TOE in this ST includes the Card Manager.

O.CARD-MANAGEMENT
The card manager shall control the access to card management functions such as the installation, update, extradition or deletion of applets and GP registry updates. It shall also implement the card issuer's policy on the card.
The card manager is an application with specific rights, which is responsible for the administration of the smart card. This component will in practice be tightly connected with the TOE, which in turn shall very likely rely on the card manager for the effective enforcing of some of its security functions. Typically the card manager shall be in charge of the life cycle of the whole card, as well as that of the installed

applications (applets). The card manager should prevent that card content management (loading, installation, deletion) is carried out, for instance, at invalid states of the card or by non-authorized actors. It shall also enforce security policies established by the card issuer.

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 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 be associated with an electronic signature (GlobalPlatform token) verified by the ISD before execution.

O.APPLI-AUTH
The card manager shall enforce the application security policies established by the card issuer by requiring application authentication during application loading on the card. This security objective is a refinement of the Security Objective O.LOAD from [PP-JCS-Open].

Application notes:

- Each application loaded onto the TOE has to be trusted.
- Each application developed by a third party, not managed by a trusted actor, has to be signed by a VA. The VA will guarantee that the security policies established by the card issuer on applications are enforced. This authority can be present on the TOE as a Security Domain whose role is to verify each signature at application loading.
- Before their loading, secure applications are previously certified by an accredited ITSEF.

O.DOMAIN-RIGHTS
The Card issuer shall not get access or change personalized AP Security Domain keys which belong to the AP. Modification of a Security Domain keyset is restricted to the AP who owns the security domain.

Application note:
APs have a set of keys that allows them to establish a secure channel between them and the platform. These keys sets are not known by the TOE issuer. The security domain initial keys are changed before any operation on the SD (OE.KEY-CHANGE).

O.COMM_AUTH
The TOE shall authenticate the origin of the card management requests that the card receives, and authenticate itself to the remote actor.

O.COMM_INTEGRITY
The TOE shall verify the integrity of the card management requests that the card receives.

O.COMM_CONFIDENTIALITY
The TOE shall be able to process card management requests containing encrypted data.

6.2.3 OS Update, OS Configurability and Secure API

The Objectives described in this section are Objectives for the additional code loading, configurability and Secure API features.

O.CONFID-OS-UPDATE.LOAD
The TOE shall be able to decrypt the additional code received for loading and installation.
The following Security Objectives have been added to comply to JIL "Security requirements for post-delivery code loading" [JIL-SECREQ].

O.Secure_Load_ACode
The TOE shall check an evidence of authenticity and integrity of the additional code to be loaded.
The TOE enforces that only an allowed version of the additional code can be loaded. The TOE shall forbid the loading of an additional code not intended to be assembled with the TOE.
During the loading of the additional code, the TOE shall remain secure.

O.Secure_AC_Activation
Activation of the additional code and update of the Identification Data shall be performed at the same time in an atomic way. All the operations needed for the code to be able to operate as in the Updated TOE shall be completed before activation.
If the atomic activation is successful, then the resulting product is the Updated TOE, otherwise (in case of interruption or incident which prevents the forming of the Updated TOE), the TOE shall remain in its initial state or fail securely.

O.TOE_Identification
The TOE provides means to store Identification Data in its non-volatile memory and guarantees the integrity of these data.
After atomic activation of the additional code, the Identification Data of the Updated TOE allows identifications of both the Initial TOE and additional code.
The user must be able to uniquely identify Initial TOE and additional code(s) which are embedded in the Updated TOE.

The Objectives described in this section are Objectives for the OS configurability feature.

O.REMOTE_SERVICE_ACTIVATION
The TOE shall perform remote optional platform service activation only when service activation is authorized and only by an authorized actor. Limited to [GemActivate Administrator (usually Thales)] under control of [OEM].

O.REMOTE_SERVICE_AUDIT
The TOE shall perform remote service audit only when optional platform service audit is authorized and only by an authorized actor. Limited to [OEM or GemActivate Administrator (usually Thales)].

The Objectives described in this section are Objectives for the Secure API.

O.Secure_API
The TOE shall provide a dedicated API - named Secure API - to applications, so as to optimize control on their sensitive operations. The Secure API shall provide security services such as secure array management, loss of data integrity detection, inconsistent execution flow detection, reaction against tearing or fault induction.

O.JCAPI-Services
The TOE shall ensure that data manipulated during SHA services as defined in [JCAPI301] cannot be observed.

6.2.4 Global Privacy Framework

This section describes the security objectives for the TOE addressing the aspects of identified threats to be countered by the Global Privacy Protocol of TOE and organisational security policies to be met by the Global Privacy Protocol of TOE.
Note: TOE objectives naming rules for this module (OT.X) is coming from [PACEPP] and remains unchanged for compatibility reason.

6.2.4.1 Security objectives for the TOE from PACE and EAC2

OT.AC_Pers_EAC2 Personalization of the TOE and Applicative data
The TOE must ensure that user data and TSF-Data that are permanently stored in the TOE can be written by authorized personalization agents only, with the following exception: An EAC2 terminal may also write or modify user data according to its effective access rights. The access rights are determined by the electronic document during Terminal Authentication 2.
Justification: This security objective for the TOE modifies OT.AC_Pers from [PACEPP] as the additional features of EAC2 allow a strongly controlled, secure and fine-grained access to individual data groups of the electronic document.

OT.Data_Integrity Integrity of Data
The TOE must ensure integrity of the User Data and the TSF-data stored on it by protecting these data against unauthorised modification (physical manipulation and unauthorised modifying).The TOE must ensure integrity of the User Data and the TSF-data during their exchange between the TOE and the terminal connected (and represented by PACE authenticated BIS-PACE) after the PACE Authentication.

OT.Data_Authenticity Authenticity of Data
The TOE must ensure authenticity of the User Data and the TSF-data stored on it by enabling verification of their authenticity at the terminal-side[2].The TOE must ensure authenticity of the User Data and the TSF-data during their exchange between the TOE and the terminal connected (and represented by PACE authenticated BIS-PACE) after the PACE Authentication. It shall happen by enabling such a verification at the terminal-side (at receiving by the terminal) and by an active verification by the TOE itself (at receiving by the TOE).

OT.Data_Confidentiality Confidentiality of Data
The TOE must ensure confidentiality of the User Data and the TSF data by granting read access only to the PACE authenticated BIS-PACE connected. The TOE must ensure confidentiality of the User Data and the TSF-data during their exchange between the TOE and the terminal connected (and represented by PACE authenticated BIS-PACE) after the PACE Authentication.

OT.Identification Identification of the TOE
The TOE must provide means to store Initialisation and Pre-Personalisation Data in its non-volatile memory. The Initialisation Data must provide a unique identification of the IC during the manufacturing and the card issuing life cycle phases of the application data requiring PACE usage (e.g. travel document for MRTD). The storage of the Pre-Personalisation data includes writing of the Personalisation Agent Key(s).

OT.Prot_Abuse_Func Protection against Abuse of Functionality
The TOE must prevent that functions of the TOE, which may not be used in TOE operational phase, can be abused in order (i) to manipulate or to disclose the User Data stored in the TOE, (ii) to manipulate or to disclose the TSF-data stored in the TOE, (iii) to manipulate (bypass, deactivate or modify) soft-coded security functionality of the TOE.

OT.Prot_Inf_Leak Protection against Information Leakage
The TOE must provide protection against disclosure of confidential User Data or/and TSF-data stored and/or processed by the TOE

- by measurement and analysis of the shape and amplitude of signals or the time between events found by measuring signals on the electromagnetic field, power consumption, clock, or I/O lines,
- by forcing a malfunction of the TOE and/or
- by a physical manipulation of the TOE.

Application note: This objective pertains to measurements with subsequent complex signal processing due to normal operation of the TOE or operations enforced by an attacker.

OT.Prot_Phys_Tamper Protection against Physical Tampering
The TOE must provide protection of confidentiality and integrity of the User Data, the TSF-data and the TOE's Embedded Software by means of

- measuring through galvanic contacts representing a direct physical probing on the chip's surface except on pads being bonded (using standard tools for measuring voltage and current) or
- measuring not using galvanic contacts, but other types of physical interaction between electrical charges (using tools used in solid-state physics research and IC failure analysis),
- manipulation of the hardware and its security functionality, as well as
- controlled manipulation of memory contents (User Data, TSF-data)
- with a prior reverse-engineering to understand the design and its properties and functionality.

OT.Prot_Malfunction Protection against Malfunctions
The TOE must ensure its correct operation. The TOE must prevent its operation outside the normal operating conditions where reliability and secure operation have not been proven or tested. This is to prevent functional errors in the TOE. The environmental conditions may include external energy (esp. electromagnetic) fields, voltage (on any contacts), clock frequency or temperature.

6.2.4.2 Security objectives for the TOE from EAC2

OT.Sens_Data_EAC2 Confidentiality of sensitive User Data
The TOE must ensure confidentiality of sensitive user data by granting access to sensitive data only to EAC2 terminals with corresponding access rights. The authorization of an EAC2 terminal is the minimum set of the access rights drawn from the terminal certificate used for successful authentication and the corresponding DV and CVCA certificates, and the access rights sent to the electronic document as part of PACE.
The TOE must ensure confidentiality of all user data during transmission to an EAC2 terminal after Chip Authentication 2. Confidentiality of sensitive user data shall be protected against attacks with high attack potential.

6.3 SECURITY OBJECTIVES FOR THE OPERATIONAL ENVIRONMENT

6.3.1 Security Objectives for the Operational Environment from Java Card System Protection Profile – Open Configuration

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

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 recommendations, if any, mandated in the platform guidance for maintaining the isolation property of the platform.
Application 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 are performed. On-card bytecode verifier is out of the scope of this Protection Profile.
Application 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.

6.3.2 Additional security objectives for the operational environment

The following security objectives for the operational environment shall also be considered for the present evaluation:

6.3.2.1 Card Management

OE.SECURITY-DOMAINS
Security domains can be dynamically created, deleted and blocked during usage phase in post-issuance mode.

OE.QUOTAS
Security domains and applets instances are subject to quotas of memory at creation and during their life time.

OE.KEY-CHANGE
The security domain keys of the VA must be securely generated prior storage in the TOE.

OE.VERIFICATION-AUTHORITY
The VA should be a trusted actor who is able to guarantee and check the digital signature attached to an application.

OE.CONTROLLING-AUTHORITY
The CA shall be a trusted actor responsible for securing the APSD keys creation and personalisation. He must be responsible for his security domain keys (CASD keys).

OE.APPS-PROVIDER
The AP shall be a trusted actor that provides basic or secure applications. He must be responsible of his security domain keys.

OE.TRUSTED-APPS-DEVELOPER
The trusted application developer shall be a trusted actor that provides basic or secure application where correct usage of the TOE has been verified applying a secure development process in a secure development environment.

OE.TRUSTED-APPS_PRE-ISSUANCE-LOADING
The pre-issuance loading on the platform must be done only using trusted or verified applets, and applying an audited process in a secure environment.

OE.GEMACTIVATE-ADMIN
The GemActivate administrator (Thales) shall be a trusted actor responsible for additional code loading/activation and optional platform service activation in post issuance.

6.3.2.2 0S Update

OE.OS-UPDATE-EVIDENCE
For additional code loaded pre-issuance, evaluated technical measures implemented by the TOE or audited organizational measures must ensure that the additional code (1) has been issued by the genuine OS Developer (2) has not been altered since it was issued by the genuine OS Developer.
For additional code loaded post-issuance, the OS Developer shall provide digital evidence to the TOE that (1) he is the genuine developer of the additional code and (2) the additional code has not been modified since it was issued by the genuine OS Developer.

OE.OS-UPDATE-ENCRYPTION
For additional code loaded post-issuance, the OS Developer shall encrypt the additional code so that its confidentiality is ensured when it is transmitted to the TOE for loading and installation.

OE.Secure_ACode_Management
Key management processes related to the OS Update capability shall take place in a secure and audited environment. The key generation processes shall guarantee that cryptographic keys are of sufficient quality and appropriately secured to ensure confidentiality, authenticity and integrity of the keys.

6.3.2.3 Global Privacy Framework

The following TOE security objectives address the aspects of identified threats to be countered involving TOE's environment.
Other security objectives for Operational environment from [EAC2PP] are specific to travel document and are not copied here.

6.3.2.3.1 Security Objectives for the Operational Environment from PACE and EAC2

OE.Personalisation Personalisation of TOE and application data requiring PACE usage
The Issuer must ensure that the Personalisation Agents acting on his behalf (i) establish the correct identity of the applicative user (e.g. travel document holder) and create the accurate applicative data and write them in TOE.
Note: in the specific case of MRTD, accurate applicative data are biographical data for the travel document), (ii) biometric reference data of the travel document holder, the initial TSF data, (the Document Security Object defined in [PKI] (in the role of a DS).

OE.Terminal Terminal operating
The terminal operators must operate their terminals as follows:

1.) The related terminals (basic inspection systems, cf. above) are used by terminal operators and by application users (e.g.travel document presenter for MRTD) as defined in [PKI].
2.) The related terminals implement the terminal parts of the PACE protocol. The PACE terminal uses randomly and (almost) uniformly selected nonces, if required by the protocols (for generating ephemeral keys for Diffie-Hellmann).
3.) The related terminals need not to use any own credentials.

4.) The related terminals and their environment must ensure confidentiality and integrity of respective data handled by them (e.g. confidentiality of the PACE passwords, integrity of PKI certificates, etc.), where it is necessary for a secure operation of the TOE according to the current ST.

OE.Prot_Logical_Data Protection of TOE and applicative data
The inspection system of the applicative entity (e.g. receiving State or Organisation) ensures the confidentiality and integrity of the data read from the TOE and applicative data (e.g. logical travel document). The inspection system will prevent eavesdropping to their communication with the TOE before secure messaging is successfully established.

OE.User_Obligations User Obligations
The application user (e.g. travel document holder) may reveal, if necessary, his or her verification values of the PACE password to an authorized person or device who definitely act according to respective regulations and are trustworthy.
Note: similar as OE.Travel_Document_Holder defined in [PACEPP] but not limited to specific application.

6.3.2.3.2 Security Objectives for the Operational Environment from EAC2

The TOE does not support pseudonymous identification and thus does not implement the Restricted Identity protocol. Consequently, the objective **OE.RestrictedIdentity** (Restricted Identity and Sector's Static Key Pairs) is not present.

OE.Chip_Auth_Key Key Pairs needed for Chip Authentication and Restricted Identification
The electronic document issuer has to ensure that the electronic document's chip authentication key pair and the Restricted Identification key pair are generated securely, that the private keys of these key pairs are stored correctly in the electronic document's chip, and that the corresponding public keys are distributed to the EAC2 terminals that are used according to [TR03110-2] to check the authenticity of the electronic document's chip.

Justification: The TSF of [PACEPP] does not include any mechanism to verify the authenticity of an electronic document (i.e. protection against cloning). Therefore, this additional security objective for the operational environment does not mitigate any threat of, and does not fulfill any OSP of [PACEPP].

OE.Terminal_Authentication Key pairs needed for Terminal Authentication
The electronic document issuer shall establish a public key infrastructure for the card verifiable certificates used for Terminal Authentication. For this aim, the electronic document issuer shall run a Country Verifying Certification Authority. The instances of the PKI shall fulfill the requirements and rules of the corresponding certificate policy. The electronic document issuer shall make the CVCA certificate available to the personalization agent or the manufacturer.

Justification: The TSF of [PACEPP] does not include any mechanism to verify the authenticity of the terminal that reads out the data stored on the electronic document (by successfully executing PACE, a terminal only proves knowledge of the PACE password). Therefore, this additional security objective for the operational environment does not mitigate any threat of, and does not fulfill any OSP of [PACEPP].

6.4 SECURITY OBJECTIVES RATIONALE

6.4.1 Security objectives rationale from Java Card System Protection Profile – Open Configuration and extension GP, OS Update, OS Configurability

6.4.1.1 Threats, OSPs and Assumptions coverage – Mapping tables

	T.PHYSICAL	T.COM_EXPLOIT	T.UNAUTHORIZED_CARD_MNGT	T.LIFE_CYCLE	T.CONFID-OS-UPDATE_LOAD	T.CONFID-APPLI-DATA	T.CONFID-JCS-DATA	T.CONFID-JCS-CODE	T.INTEG-APPLI-CODE	T.INTEG-APPLI-CODE.LOAD	T.INTEG-APPLI-DATA	T.INTEG-APPLI-DATA.LOAD	T.INTEG-JCS-DATA	T.INTEG-JCS-CODE	T.SID.1	T.SID.2	T.EXE-CODE.1	T.EXE-CODE.2	T.NATIVE	T.RESOURCES	T.DELETION	T.INSTALL	T.OBJ-DELETION	T.UNAUTHORIZED_ACCESS_TO_SERVICE	T.UNAUTHORIZED_TOE_CODE_UPDATE	T.WRONG-UPDATE-STATE	T.INTEG-OS-UPDATE_LOAD	T.FAKE-CERT
O.CARD-MANAGEMENT			X	X		X	X	X	X	X	X	X	X	X	X						X	X						
O.DOMAIN-RIGHTS			X	X																								
O.APPLI-AUTH			X																									
O.COMM_AUTH		X	X																									
O.COMM_INTEGRITY		X	X																									
O.COMM_CONFIDENTIALITY		X																										
O.SCP-SUPPORT	X					X	X				X		X			X				X								
O.SID						X	X				X		X		X	X												
O.FIREWALL						X	X				X		X		X	X	X											
O.GLOBAL_ARRAYS_CONFID						X									X													
O.GLOBAL_ARRAYS_INTEG											X				X													
O.NATIVE								X	X					X					X									
O.OPERATE						X	X				X		X			X				X								
O.REALLOCATION						X					X																	
O.RESOURCES																				X								
O.ALARM						X	X				X		X															
O.CIPHER						X					X																	
O.KEY-MNGT						X					X																	
O.PIN-MNGT						X					X																	
O.TRANSACTION						X					X																	
O.OBJ-DELETION																							X					
O.DELETION																					X							
O.LOAD										X		X										X						
O.INSTALL															X	X				X		X						
O.SCP.IC	X																											
O.SCP.RECOVERY						X	X				X		X			X				X								
O.RNG						X					X																	
O.Secure_API																												
O.JCAPI-Services																												
O.REMOTE_SERVICE_AUDIT																												
O.REMOTE_SERVICE_ACTIVATION																								X				
O.CONFID-OS-UPDATE.LOAD					X																							
O.Secure_Load_ACode																									X		X	
O.Secure_AC_Activation																									X	X		
O.TOE_Identification																											X	X
OE.APPS-PROVIDER																												

OE.VERIFICATION-AUTHORITY																									
OE.CONTROLLING-AUTHORITY																									
OE.KEY-CHANGE																									
OE.SECURITY-DOMAINS																									
OE.QUOTAS																									
OE.APPLET																	X								
OE.VERIFICATION						X	X	X	X		X		X	X			X	X	X							
OE.CODE-EVIDENCE									X	X	X	X	X													
OE.TRUSTED-APPS-DEVELOPER																									
OE.TRUSTED-APPS_PRE-ISSUANCE-LOADING																									
OE.GEMACTIVATE-ADMIN																						X			
OE.OS-UPDATE-EVIDENCE																									
OE.OS-UPDATE-ENCRYPTION					X																				
OE.Secure_ACode_Management																							X	X		X

Table 7: Threats coverage by security objectives – Mapping table

	OSP.VERIFICATION	OSP.KEY-CHANGE	OSP.SECURITY-DOMAINS	OSP.QUOTAS	OSP.TRUSTED-APPS-DEVELOPER	OSP.TRUSTED-APPS_PRE-ISSUANCE-LOADING	OSP.SERVICE_AUDIT	OSP.ACTIVATION-KEY-ACTOR	OSP.ATOMIC_ACTIVATION	OSP.TOE_IDENTIFICATION	OSP.ADDITIONAL_CODE_ENCRYPTION	OSP.ADDITIONAL_CODE_SIGNING	OSP.SecureAPI	OSP.JCAPI-Services
O.CARD-MANAGEMENT														
O.DOMAIN-RIGHTS														
O.APPLI-AUTH														
O.COMM_AUTH														
O.COMM_INTEGRITY														
O.COMM_CONFIDENTIALITY														
O.SCP-SUPPORT														
O.SID														
O.FIREWALL														
O.GLOBAL_ARRAYS_CONFID														
O.GLOBAL_ARRAYS_INTEG														
O.NATIVE														
O.OPERATE														
O.REALLOCATION														
O.RESOURCES														
O.ALARM														
O.CIPHER														
O.KEY-MNGT														
O.PIN-MNGT														
O.TRANSACTION														
O.OBJ-DELETION														
O.DELETION														
O.LOAD	X													
O.INSTALL														
O.SCP.IC														
O.SCP.RECOVERY														
O.RNG														
O.Secure_API													X	
O.JCAPI-Services														X
O.REMOTE_SERVICE_AUDIT							X							

O.REMOTE_SERVICE_ACTIVATION											X			
O.CONFID-OS-UPDATE.LOAD									X			X		
O.Secure_Load_ACode									X					
O.Secure_AC_Activation										X				
O.TOE_Identification														
OE.APPS-PROVIDER														
OE.VERIFICATION-AUTHORITY														
OE.CONTROLLING-AUTHORITY														
OE.KEY-CHANGE		X												
OE.SECURITY-DOMAINS			X											
OE.QUOTAS				X										
OE.APPLET														
OE.VERIFICATION	X													
OE.CODE-EVIDENCE	X													
OE.TRUSTED-APPS-DEVELOPER					X									
OE.TRUSTED-APPS_PRE-ISSUANCE-LOADING						X								
OE.GEMACTIVATE-ADMIN														
OE.OS-UPDATE-EVIDENCE								X						
OE.OS-UPDATE-ENCRYPTION											X			
OE.Secure_ACode_Management												X		

Table 8: OSP coverage by security objectives – Mapping table

	A.APPS-PROVIDER	A.VERIFICATION-AUTHORITY	A.CONTROLLING-AUTHORITY	A.APPLET	A.VERIFICATION	A.DELETION
O.CARD-MANAGEMENT						X
O.DOMAIN-RIGHTS						
O.APPLI-AUTH						
O.COMM_AUTH						
O.COMM_INTEGRITY						
O.COMM_CONFIDENTIALITY						
O.SCP-SUPPORT						
O.SID						
O.FIREWALL						
O.GLOBAL_ARRAYS_CONFID						
O.GLOBAL_ARRAYS_INTEG						
O.NATIVE						
O.OPERATE						
O.REALLOCATION						
O.RESOURCES						
O.ALARM						
O.CIPHER						
O.KEY-MNGT						
O.PIN-MNGT						
O.TRANSACTION						
O.OBJ-DELETION						
O.DELETION						
O.LOAD						
O.INSTALL						
O.SCP.IC						
O.SCP.RECOVERY						
O.RNG						
O.Secure_API						
O.JCAPI_Services						
O.REMOTE_SERVICE_AUDIT						

O.REMOTE_SERVICE_ACTIVATION						
O.CONFID-OS-UPDATE.LOAD						
O.Secure_Load_ACode						
O.Secure_AC_Activation						
O.TOE_Identification						
OE.APPS-PROVIDER	X					
OE.VERIFICATION-AUTHORITY		X				
OE.CONTROLLING-AUTHORITY			X			
OE.KEY-CHANGE						
OE.SECURITY-DOMAINS						
OE.QUOTAS						
OE.APPLET				X		
OE.VERIFICATION					X	
OE.CODE-EVIDENCE					X	
OE.TRUSTED-APPS-DEVELOPER						
OE.TRUSTED-APPS_PRE-ISSUANCE-LOADING						
OE.GEMACTIVATE-ADMIN						
OE.OS-UPDATE-EVIDENCE						
OE.OS-UPDATE-ENCRYPTION						
OE.Secure_ACode_Management						

Table 9: Assumptions coverage by security objectives – Mapping table

6.4.1.2 Threats coverage – Rationale

T.PHYSICAL
This threat is countered by physical protections which rely on the underlying platform. The security objectives O.SCP-SUPPORT and O.SCP.IC protect sensitive assets of the platform against loss of integrity and confidentiality and especially ensure the TSFs cannot be bypassed or altered.

T.COM_EXPLOIT
This threat is covered by the following security objectives:

- O.COMM_AUTH prevents unauthorized users from initiating a malicious card management operation.
- O.COMM_INTEGRITY protects the integrity of the card management data while it is in transit to the TOE.
- O.COMM_CONFIDENTIALITY prevents from disclosing encrypted data transiting to the TOE.

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.COMM_AUTH prevents unauthorized users from initiating a malicious card management operation.
- O.COMM_INTEGRITY protects the integrity of the card management data while it is in transit to the TOE.
- O.APPLI-AUTH which requires for loading all applications to be authenticated.
- O.DOMAIN-RIGHTS which restricts the modification of an AP security domain keyset to the AP who owns it.

T.LIFE_CYCLE
This threat is covered by the security objectives:

- 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
- O.DOMAIN-RIGHTS that restricts the use of an AP security domain keysets, and thus the management of the applications related to this SD, to the AP who owns it.

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 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.
- 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, O.RNG). 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-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 objective 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.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 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.

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. The (#.VERIFICATION) security aspect is addressed in this configuration by the objective for the environment OE.VERIFICATION.
- Native applications are also harmless because of the objectives O.NATIVE so no application can be run to modify a piece of code.
- 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.
- 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, O.RNG). 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-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.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.

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 objectives OE.VERIFICATION and OE.BASIC-APPS-VALIDATION. 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.

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

- This threat is directly countered by objectives on resource-management (O.RESOURCES) for runtime purposes and good working order (O.OPERATE) in a general manner.
- Consumption of resources during installation and other card management operations are covered, in case of failure, by O.INSTALL.
- Finally, the objectives O.SCP.RECOVERY and O.SCP-SUPPORT are intended to support the O.OPERATE and O.RESOURCES objectives of the TOE, so they are indirectly related to the threats that these latter objectives contribute to counter.

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.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.UNAUTHORIZED_ACCESS_TO_SERVICE
This threat is countered by the security objectives O.REMOTE_SERVICE_ACTIVATION and OE.GEMACTIVATE-ADMIN where only an authorized and trusted actor is able to activate optional services.

T.UNAUTHORIZED_TOE_CODE_UPDATE
This threat is countered by the security objectives O.Secure_Load_ACode, O.Secure_AC_Activation and OE.Secure_ACode_Management.

T.WRONG-UPDATE-STATE
This threat is countered by the security objectives O.Secure_AC_Activation and OE.Secure_ACode_Management.

T. INTEG-OS-UPDATE_LOAD
This threat is countered by the security objectives O.Secure_Load_ACode and O.TOE_Identification.

T.CONFID-OS-UPDATE_LOAD
This threat is countered by the security objectives O.CONFID-OS-UPDATE.LOAD and OE.OS-UPDATE-ENCRYPTION.

T.FAKE-CERT
This threat is countered by the security objectives O.TOE_Identification and OE.Secure_ACode_Management.

6.4.1.3 OSP coverage – Rationale

OSP.KEY-CHANGE
This OSP is enforced by the security objective for the operational environment of the TOE OE.KEY CHANGE.

OSP.SECURITY-DOMAINS
This OSP is enforced by the security objective for the operational environment of the TOE OE.SECURITY-DOMAINS.

OSP.QUOTAS
This OSP is enforced by the security objective for the operational environment of the TOE OE.QUOTAS.

OSP.VERIFICATION
- This policy is upheld by the security objectives 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, and by the security objective for the TOE O.LOAD which shall ensure that the loading of a package into the card is safe.

OSP.SecureAPI
This OSP is enforced by the TOE security objective O.Secure_API.

OSP.JCAPI-Services
This OSP is enforced by the TOE security objective O.JCAPI-Services.

OSP.TRUSTED-APPS-DEVELOPER
This OSP is enforced by the security objective OE.TRUSTED-APPS-DEVELOPER.

OSP.TRUSTED-APPS_PRE-ISSUANCE-LOADING

This OSP is enforced by the security objective OE.TRUSTED-APPS_PRE-ISSUANCE-LOADING.

OSP.SERVICE_AUDIT
This OSP is directly enforced by the security objective O.REMOTE_SERVICE_AUDIT.

OSP.ACTIVATION-KEY-ACTOR
This OSP is enforced by the security objective OE.OS-UPDATE-EVIDENCE.

OSP.Atomic_Activation
This OSP is enforced by the security objective O.Secure_AC_Activation and O.Secure_Load_ACode.

OSP.TOE_Identification
This OSP is enforced by the security objective O.TOE_Identification.

OSP.Additional_Code_Signing
This OSP is enforced by the security objectives O.Secure_Load_ACode and OE.Secure_ACode_Management.

OSP.Additional_Code_Encryption
This OSP is enforced by the security objectives O.CONFID-OS-UPDATE.LOAD and OE.OS-UPDATE-ENCRYPTION.

6.4.1.4 Assumptions coverage – Rationale

A.APPS-PROVIDER
This assumption is directly upheld by OE.APPS-PROVIDER.

A.VERIFICATION-AUTHORITY
This assumption is directly upheld by OE.VERIFICATION-AUTHORITY.

A.CONTROLLING-AUTHORITY
This assumption is directly upheld by OE.CONTROLLING-AUTHORITY.

A.APPLET
This assumption is upheld by the security objective for the operational environment OE.APPLET which ensures that no applet loaded post-issuance shall contain native methods.

A.VERIFICATION
This assumption is upheld by the security objective on the operational 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 assumption 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.

A.DELETION
The assumption A.DELETION is upheld by the security objective O.CARD-MANAGEMENT which controls the access to card management functions such as deletion of applets.

6.4.2 Security objectives rationale for Global Privacy Framework

6.4.2.1 Threats

The following table provides an overview for security objectives coverage.

	OT.AC_Pers_EAC2	*OT.Data_Integrity*	*OT.Data_Authenticity*	*OT.Data_Confidentiality*	*OT.Identification*	*OT.Prot_Abuse-Func*	*OT.Prot_Inf_Leak*	*OT.Prot_Phys-Tamper*	*OT.Prot_Malfuntion*	*OT.Sens_Data_EAC2*	OE.Prot_Logical_Data	*OE.Personalisation*	*OE.Terminal*	OE.User_Obligations	*OE.Chip_Auth_Key*	*OE.Terminal_Authentication*
T.Skimming[3]		X	X	X						X				X		X
T.Eavesdropping				X						X						
T.Abuse-Func						X										
T.Information_Leakage							X									
T.Phys-Tamper								X								
T.Malfunction									X							
T.Forgery	X	X	X			X		X				X	X			
T.Sensitive_Data										X						X

Table 10: Threats vs Security Objectives for Privacy Framework

The threat **T.Skimming** addresses accessing the user data (stored on the TOE or transferred between the TOE and the terminal) using the TOE's contactless/contact-based interface. This threat is countered by the security objectives **OT.Data_Integrity, OT.Data_Authenticity** and **OT.Data_Confidentiality** through the PACE authentication. The objective **OE.User_Obligations** ensures that a PACE session can only be established either by the application user itself (e.g. travel document holder for MRTD) or by an authorised person or device, and, hence, cannot be captured by an attacker. Additionally to the security objectives from [PACEPP] which counter this threat, the threat is also addressed by **OT.Sens_Data_EAC2** that demands a trusted channel based on Chip Authentication 2, and requires that read access to sensitive user data is only granted to EAC2 terminals with corresponding access rights. Moreover, **OE.Terminal_Authentication** requires the electronic document issuer to provide the corresponding PKI.

The threat **T.Eavesdropping** addresses listening to the communication between the TOE and a PACE terminal or an EAC2 terminal in order to gain access to transferred user data. This threat is countered by the security objective **OT.Data_Confidentiality** through a trusted channel based on PACE Authentication, and by **OT.Sens_Data_EAC2** demanding a trusted channel that is based on Chip Authentication 2.

The threat **T.Abuse-Func** addresses attacks of misusing TOE's functionality to manipulate or to disclosure the stored User- or TSF-data as well as to disable or to bypass the soft-coded security functionality. The security objective **OT.Prot_Abuse-Func** ensures that the usage of functions having not to be used in the operational phase is effectively prevented.

[3] Threats and assumptions included from [EAC2PP] are marked *in italic letters.* They are listed for the complete overview of threats and assumptions.

The threats **T.Information_Leakage**, **T.Phys-Tamper** and **T.Malfunction** are typical for integrated circuits like smart cards under direct attack with high attack potential. The protection of the TOE against these threats is obviously addressed by the directly related security objectives **OT.Prot_Inf_Leak, OT.Prot_Phys-Tamper** and **OT.Prot_Malfunction**, respectively.

The threat **T.Forgery** addresses the fraudulent, complete or partial alteration of user data and/or TSF-Data stored on the TOE, and/or exchanged between the TOE and the terminal. The security objective **OT.AC_Pers_EAC2** requires the TOE to limit the write access for the TOE and applicative data to the trustworthy Personalisation Agent (cf. **OE.Personalisation**). The TOE will protect the integrity and authenticity of the stored and exchanged User Data or/and TSF-data as aimed by the security objectives **OT.Data_Integrity** and **OT.Data_Authenticity**, respectively. The objectives **OT.Prot_Phys-Tamper** and **OT.Prot_Abuse-Func** contribute to protecting integrity of the User Data or/and TSF-data stored on the TOE. A terminal operator operating his terminals according to **OE.Terminal** to contribute to secure exchange between the TOE and the terminal.

The threat **T.Sensitive_Data** is countered by the TOE-Objective **OT.Sens_Data_EAC2** that requires that read access to sensitive user data is only granted to EAC2 terminals with corresponding access rights. Furthermore, it is required that the confidentiality of the data is ensured during transmission. The objective **OE.Terminal_Authentication** requires the electronic document issuer to provide the public key infrastructure (PKI) to generate and distribute the card verifiable certificates needed by the electronic document to securely authenticate the EAC2 terminal.

6.4.2.2 *Organizational Security Policies and Assumptions*

	OT.AC_Pers_EAC2	*OT.Data_Integrity*	*OT.Data_Authenticity*	*OT.Data_Confidentiality*	*OT.Identification*	*OT.Prot_Abuse-Func*	*OT.Prot_Inf_Leak*	*OT.Prot_Phys-Tamper*	*OT.Prot_Malfuntion*	*OT.Sens_Data_EAC2*	OE.Prot_Logical_Data	*OE.Personalisation*	*OE.Terminal*	OE.User_Obligations	*OE.Chip_Auth_Key*	*OE.Terminal_Authentication*
P.Personalisation	X				X							X				
P.Manufact					X											
P.Pre-Operational	X				X							X				
P.Terminal													X			
P.EAC2_Terminal													X		X	X
A.Insp_Sys											X					

Table 11: OSP and Assumptions vs Security Objectives for Privacy Framework

The OSP **P.Personalisation** addresses the (i) the enrolment of the logical travel document by the Personalisation Agent as described in the security objective for the TOE environment **OE.Personalisation**, and (ii) the access control for the user data and TSF data as described by the security objective **OT.AC_Pers.** Note the manufacturer equips the TOE with the Personalisation Agent Key(s) according to **OT.Identification** “Identification and Authentication of the TOE”.

The OSP **P.Manufact** requires a unique identification of the IC by means of the Initialization Data and the writing of the Pre-personalisation Data as being fulfilled by **OT.Identification**.

The OSP **P.Pre-Operational** is enforced by the following security objectives: **OT.Identification** is affine to the OSP's property 'traceability before the operational phase'; **OT.AC_Pers_EAC2** and **OE.Personalisation** together enforce the OSP's properties 'correctness of the User- and the TSF-data stored' and 'authorisation of Personalisation Agents'.

The OSP **P.Terminal** "Abilities and trustworthiness of terminals" is countered by the security objective **OE.Terminal** enforces the terminals to perform the terminal part of the PACE protocol.

The OSP **P.EAC2_Terminal** addresses the requirement for EAC2 terminals to implement the terminal parts of the protocols needed to executed EAC2 according to its specification in [TR03110-2]: , and to store (static keys) or generate (temporary keys and nonces) the needed related credentials. This is enforced by **OE.Chip_Auth_Key** which requires Chip Authentication and Restricted Identity keys to be correctly generated and stored, by **OE.Terminal_Authentication** for the PKI needed for Terminal Authentication, and by **OE.Terminal** which covers the PACE protocol and the Passive Authentication protocol.

A.Insp_Sys is covered by OE.Prot_Logical_Data requiring the Inspection System to protect the TOE and application data (e.g. the logical travel document data) during the transmission and the internal handling.

6.4.3 Compatibility between Security Objectives of [ST-JCS] and [ST-IC]

6.4.3.1 Compatibility between objectives for the TOE

The following table lists the relevant TOE security objectives of the IC, and provides the link to the composite-product TOE security objectives, showing that there is no contradiction between the two sets of objectives.

Label of the chip TOE security objective	Title of the chip TOE security objective	Content of the chip TOE security objective	Linked Composite-product TOE security objectives
O.Leak-Inherent	Protection against Inherent Information Leakage	The TOE must provide protection against disclosure of confidential data stored and/or processed in the Security IC - by measurement and analysis of the shape and amplitude of signals (for example on the power, clock, or I/O lines) and - by measurement and analysis of the time between events found by measuring signals (for instance on the power, clock, or I/O lines). This objective pertains to measurements with subsequent complex signal processing whereas O.Phys-Probing is about direct measurements on elements on the chip surface.	O.SCP-SUPPORT O.SCP.IC
O.Phys-Probing	Protection against Physical Probing	The TOE must provide protection against disclosure of User Data, against the disclosure/reconstruction of the Security IC Embedded Software or against the disclosure of other critical information about the operation of the TOE. This includes protection against - measuring through galvanic contacts which is direct physical probing on the chips surface except on pads being bonded (using standard tools for measuring voltage and current) or - measuring not using galvanic contacts but other types of physical interaction between charges (using tools used in solid-state physics research and IC failure analysis) with a prior reverse-engineering to understand the design and its properties and functions.	O.SCP-SUPPORT O.SCP.IC
O.Malfunction	Protection against Malfunctions	The TOE must ensure its correct operation. The TOE must indicate or prevent its operation outside the normal operating conditions where reliability and secure operation has not been proven or tested. This is to prevent malfunctions. Examples of environmental conditions are voltage, clock frequency, temperature, or external energy fields.	O.OPERATE
O.Phys-Manipulation	Protection against Physical Manipulation	The TOE must provide protection against manipulation of the TOE (including its software and Data), the Security IC Embedded Software and the User Data. This includes protection against - reverse-engineering (understanding the design and its properties and functions), - manipulation of the hardware and any data, as well as - controlled manipulation of memory contents (Application Data).	O.SCP-SUPPORT O.SCP.IC
O.Leak-Forced	Protection against Forced Information Leakage	The Security IC must be protected against disclosure of confidential data processed in the Security IC (using methods as described under O.Leak-Inherent) even if the information leakage is not inherent but caused by the attacker - by forcing a malfunction (refer to “Protection against Malfunction due to Environmental Stress (O.Malfunction)” and/or	O.SCP-SUPPORT O.SCP.IC

Label of the chip TOE security objective	Title of the chip TOE security objective	Content of the chip TOE security objective	Linked Composite-product TOE security objectives
		- by a physical manipulation (refer to “Protection against Physical Manipulation (O.Phys-Manipulation)”. If this is not the case, signals which normally do not contain significant information about secrets could become an information channel for a leakage attack.	
O.Abuse-Func	Protection against Abuse of Functionality	The TOE must prevent that functions of the TOE which may not be used after TOE Delivery can be abused in order to (i) disclose critical User Data, (ii) manipulate critical User Data of the Security IC Embedded Software, (iii) manipulate Soft-coded Security IC Embedded Software or (iv) bypass, deactivate, change or explore security features or security services of the TOE. Details depend, for instance, on the capabilities of the Test Features provided by the IC Dedicated Test Software which are not specified here.	O.SCP-SUPPORT
O.Identification	TOE Identification	The TOE must provide means to store Initialization Data and Pre-personalization Data in its non-volatile memory. The Initialization Data (or parts of them) are used for TOE identification.	No direct link to the composite-product TOE objectives, however chip traceability information stored in NVM is used by the TOE to answer identification CC requirements.
O.RND	Random Numbers	The TOE will ensure the cryptographic quality of random number generation. For instance random numbers shall not be predictable and shall have a sufficient entropy. The TOE will ensure that no information about the produced random numbers is available to an attacker since they might be used for instance to generate cryptographic keys.	O.RNG
O.Mem_Access	Area based Memory Access Control	The TOE must provide the Smartcard Embedded Software with the capability to define restricted access memory areas. The TOE must then enforce the partitioning of such memory areas so that access of software to memory areas is controlled as required, for example, in a multi-application environment.	O.SCP-SUPPORT
O.Cap_Avail_Loader	Capability and availability of the Loader	The TSF provides limited capability of the Loader functionality and irreversible termination of the Loader in order to protect stored user data from disclosure and manipulation.	This IC security objective supports the loading of the UpTeq NFC422 v1.0 in term of confidentiality and integrity.
O.Ctrl_Auth_Loader	Access control and authenticity for the Loader	The TSF provides trusted communication channel with authorized user, supports confidentiality protection and authentication of the user data to be loaded and access control for usage of the Loader functionality.	This IC security objective supports the loading of the

Label of the chip TOE security objective	Title of the chip TOE security objective	Content of the chip TOE security objective	Linked Composite-product TOE security objectives
			UpTeq NFC422 v1.0 using a secured communication channel.
O.TDES	Cryptographic service Triple-DES	The TOE provides secure hardware based cryptographic services implementing the Triple-DES for encryption and decryption	O.CIPHER
O.AES	Cryptographic service AES	The TOE provides secure hardware based cryptographic services implementing the AES for encryption and decryption	O.CIPHER
O.Authentication	Authentication to external entities	The TOE shall be able to authenticate itself to external entities. The initialization Data (or parts of them) are used for TOE authentication verification data	O.SCP-SUPPORT O.SCP.IC
O.Prot_TSF_Confidentiality	Protection of the confidentiality of the TSF	The TOE must provide protection against disclosure of confidential operations of the Security IC (loader, memory management unit,…) through the use of a dedicated code loader on open samples.	O.SCP-SUPPORT

6.4.3.2 Compatibility between objectives for the Environment

The following table lists the relevant ENV security objectives related to the IC, and provides the link to the composite-product, showing that they have been taken into account and that no contradiction has been introduced.

IC ENV security objective label	IC ENV security objective title	IC ENV security objective content	Link to the composite-product
OE.Resp-Appl	Treatment of User Data of the Composite TOE	Security relevant User Data (especially cryptographic keys) are treated by the Security IC Embedded Software as required by the security needs of the specific application context.	Covered by TOE Security Objectives: O.COMM_AUTH, O.COMM_CONFIDENTIALITY O.KEY-MNGT, O.PIN-MNGT

		For example the Security IC Embedded Software will not disclose security relevant User Data to unauthorized users or processes when communicating with a terminal.	
OE.Process-Sec-IC	Protection during composite product manufacturing	Security procedures shall be used after TOE Delivery up to delivery to the end-consumer to maintain confidentiality and integrity of the TOE and of its manufacturing and test data (to prevent any possible copy, modification, retention, theft or unauthorized use). This means that Phases after TOE Delivery up to the end of Phase 6 must be protected appropriately.	• During phases 4, 5 and 6: covered by the ALC composite-SARs. • During phase 7, covered by all ENV objectives of the composite-TOE.
OE.Lim_Block_Loader	Limitation of capability and blocking the Loader	The composite Product Manufacturer will protect the Loader functionality against misuse, limit the capability of the Loader and terminate irreversibly the Loader after intended usage of the Loader and before the end of phase 5.	Protection applicable for the TOE loading. Covered by the ALC composite-SARs.
OE.Loader_Usage	Secure communication and usage of the Loader	The authorized user must support the trusted communication channel with the TOE by confidentiality protection and authenticity proof of the data to be loaded and fulfilling the access conditions required by the Loader.	Protection applicable for the TOE loading. Covered by the ALC composite-SARs.
OE.TOE_Auth	External entities authenticating of the TOE	The operational environment shall support the authentication verification mechanism and know authentication reference data of the TOE.	Protection applicable for the TOE loading. Covered by the ALC composite-SARs.

6.4.4 Compatibility between Security Objectives of Global Privacy Framework and [ST-IC]

6.4.4.1 Compatibility between objectives for the TOE

The following table lists the relevant TOE security objectives of the IC, and provides the link to the composite-product TOE Privacy Framework part security objectives, showing that there is no contradiction between the two sets of objectives.

Label of the chip TOE security objective	Title of the chip TOE security objective	Content of the chip TOE security objective	Linked Composite-product TOE security objectives
O.Leak-Inherent	Protection against Inherent Information Leakage	The TOE must provide protection against disclosure of confidential data stored and/or processed in the Security IC - by measurement and analysis of the shape and amplitude of signals (for example on the power, clock, or I/O lines) and - by measurement and analysis of the time between events found by measuring signals (for instance on the power, clock, or I/O lines). This objective pertains to measurements with subsequent complex signal processing whereas O.Phys-Probing is about direct measurements on elements on the chip surface.	OT.Prot_Inf_Leak
O.Phys-Probing	Protection against Physical Probing	The TOE must provide protection against disclosure of User Data, against the disclosure/reconstruction of the Security IC Embedded Software or against the disclosure of other critical information about the operation of the TOE. This includes protection against - measuring through galvanic contacts which is direct physical probing on the chips surface except on pads being bonded (using standard tools for measuring voltage and current) or - measuring not using galvanic contacts but other types of physical interaction between charges (using tools used in solid-state physics research and IC failure analysis) with a prior reverse-engineering to understand the design and its properties and functions.	Not applicable
O.Malfunction	Protection against Malfunctions	The TOE must ensure its correct operation. The TOE must indicate or prevent its operation outside the normal operating conditions where reliability and secure operation has not been proven or tested. This is to prevent malfunctions. Examples of environmental conditions are voltage, clock frequency, temperature, or external energy fields.	OT.Prot_Malfunction
O.Phys-Manipulation	Protection against Physical Manipulation	The TOE must provide protection against manipulation of the TOE (including its software and Data), the Security IC Embedded Software and the User Data. This includes protection against - reverse-engineering (understanding the design and its properties and functions), - manipulation of the hardware and any data, as well as - controlled manipulation of memory contents (Application Data).	OT.Data_Confidentiality OT.Data_Integrity OT.Data_Authenticity OT.Prot_Phys-Tamper
O.Leak-Forced	Protection against Forced Information Leakage	The Security IC must be protected against disclosure of confidential data processed in the Security IC (using methods as described under O.Leak-Inherent) even if the information leakage is not inherent but caused by the attacker	OT.Prot_Inf_Leak

Label of the chip TOE security objective	Title of the chip TOE security objective	Content of the chip TOE security objective	Linked Composite-product TOE security objectives
		- by forcing a malfunction (refer to “Protection against Malfunction due to Environmental Stress (O.Malfunction)” and/or - by a physical manipulation (refer to “Protection against Physical Manipulation (O.Phys-Manipulation)”. If this is not the case, signals which normally do not contain significant information about secrets could become an information channel for a leakage attack.	
O.Abuse-Func	Protection against Abuse of Functionality	The TOE must prevent that functions of the TOE which may not be used after TOE Delivery can be abused in order to (i) disclose critical User Data, (ii) manipulate critical User Data of the Security IC Embedded Software, (iii) manipulate Soft-coded Security IC Embedded Software or (iv) bypass, deactivate, change or explore security features or security services of the TOE. Details depend, for instance, on the capabilities of the Test Features provided by the IC Dedicated Test Software which are not specified here.	OT.Prot_Abuse-Func
O.Identification	TOE Identification	The TOE must provide means to store Initialization Data and Pre-personalization Data in its non-volatile memory. The Initialization Data (or parts of them) are used for TOE identification.	OT.Identification
O.RND	Random Numbers	The TOE will ensure the cryptographic quality of random number generation. For instance random numbers shall not be predictable and shall have a sufficient entropy. The TOE will ensure that no information about the produced random numbers is available to an attacker since they might be used for instance to generate cryptographic keys.	OT.Data_Confidentiality OT.Data_Integrity OT.Data_Authenticity
O.Mem_Access	Area based Memory Access Control	The TOE must provide the Smartcard Embedded Software with the capability to define restricted access memory areas. The TOE must then enforce the partitioning of such memory areas so that access of software to memory areas is controlled as required, for example, in a multi-application environment.	Not applicable
O.Cap_Avail_Loader	Capability and availability of the Loader	The TSF provides limited capability of the Loader functionality and irreversible termination of the Loader in order to protect stored user data from disclosure and manipulation.	Not applicable
O.Ctrl_Auth_Loader	Access control and authenticity for the Loader	The TSF provides trusted communication channel with authorized user, supports confidentiality protection and authentication of the user data to be loaded and access control for usage of the Loader functionality.	Not applicable
O.TDES	Cryptographic service Triple-DES	The TOE provides secure hardware based cryptographic services implementing the Triple-DES for encryption and decryption	Not applicable
O.AES	Cryptographic service AES	The TOE provides secure hardware based cryptographic services implementing the AES for encryption and decryption	Not applicable
O.Authentication	Authentication to external entities	The TOE shall be able to authenticate itself to external entities. The initialization Data (or parts of them) are used for TOE authentication verification data	OT.C_Pers_EAC2 OT.Data_Authencity

Label of the chip TOE security objective	Title of the chip TOE security objective	Content of the chip TOE security objective	Linked Composite-product TOE security objectives
O.Prot_TSF_Confidentiality	Protection of the confidentiality of the TSF	The TOE must provide protection against disclosure of confidential operations of the Security IC (loader, memory management unit,…) through the use of a dedicated code loader on open samples.	Not applicable

6.4.4.2 Compatibility between objectives for the environment

The following table lists the relevant ENV security objectives related to the IC, and provides the link to the composite-product Privacy Framework part, showing that they have been taken into account and that no contradiction has been introduced.

IC ENV security objective label	IC ENV security objective title	IC ENV security objective content	Link to the composite-product
OE.Resp-Appl	Treatment of User Data of the Composite TOE	Security relevant User Data (especially cryptographic keys) are treated by the Security IC Embedded Software as required by the security needs of the specific application context. For example the Security IC Embedded Software will not disclose security relevant User Data to unauthorized users or processes when communicating with a terminal.	OE.Personalisation
OE.Process-Sec-IC	Protection during composite product manufacturing	Security procedures shall be used after TOE Delivery up to delivery to the end-consumer to maintain confidentiality and integrity of the TOE and of its manufacturing and test data (to prevent any possible copy, modification, retention, theft or unauthorized use). This means that Phases after TOE Delivery up to the end of Phase 6 must be protected appropriately.	OE.Personalisation
OE.Lim_Block_Loader	Limitation of capability and blocking the Loader	The composite Product Manufacturer will protect the Loader functionality against misuse, limit the capability of the Loader and terminate irreversibly the Loader after intended usage of the Loader and before the end of phase 5.	Not applicable

OE.Loader_Usage	Secure communication and usage of the Loader	The authorized user must support the trusted communication channel with the TOE by confidentiality protection and authenticity proof of the data to be loaded and fulfilling the access conditions required by the Loader.	Not applicable
OE.TOE_Auth	External entities authenticating of the TOE	The operational environment shall support the authentication verification mechanism and know authentication reference data of the TOE.	Not applicable

7 SECURITY REQUIREMENTS

7.1 EXTENDED COMPONENTS DEFINITION

This security target uses components defined as extensions to CC part 2.
Some of these components are defined in protection profile [PP-JCS-Open], others are defined in the protection profile [PP-IC-0084] and [EAC2PP].

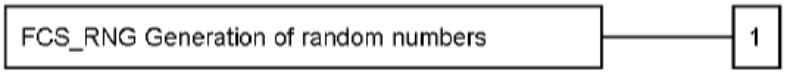
7.1.1 Definition of the Family FCS_RNG

To define the IT security functional requirements of the TOE an additional family (FCS_RNG) of the Class FCS (cryptographic support) is defined here. This family describes the functional requirements for random number generation used for cryptographic purposes.

FCS_RNG Generation of random numbers

Family behaviour
This family defines quality requirements for the generation of random numbers which are intended to be use for cryptographic purposes.

Component levelling:



FCS_RNG.1	Generation of random numbers requires that random numbers meet a defined quality metric.
Management:	FCS_RNG.1 There are no management activities foreseen.
Audit:	FCS_RNG.1 There are no actions defined to be auditable.

FCS_RNG.1 Random number generation

Hierarchical to:	No other components
Dependencies:	No dependencies

FCS_RNG.1.1	The TSF shall provide a [selection: physical, non-physical true, deterministic, hybrid physical, hybrid deterministic] random number generator that implements: [assignment: list of security capabilities].
FCS_RNG.1.2	The TSF shall provide random numbers that meet [assignment: a defined quality metric].

7.1.2 Definition of the Family FMT_LIM

The family FMT_LIM describes the functional requirements for the Test Features of the TOE. The new functional requirements were defined in the class FMT because this class addresses the management of functions of the TSF. The examples of the technical mechanism used in the TOE show that no other class is

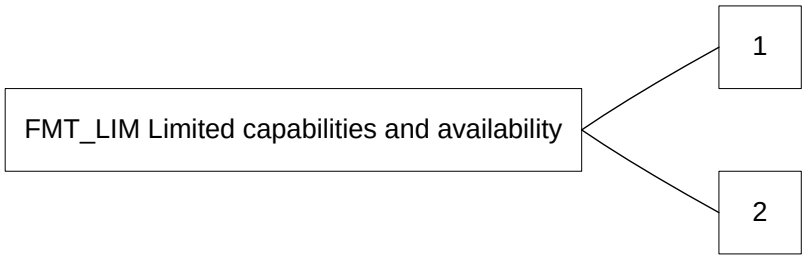
appropriate to address the specific issues of preventing the abuse of functions by limiting the capabilities of the functions and by limiting their availability.

The family “Limited capabilities and availability (FMT_LIM)” is specified as follows.

FMT_LIM Limited capabilities and availability

Family behavior
This family defines requirements that limit the capabilities and availability of functions in a combined manner. Note that FDP_ACF restricts the access to functions whereas the Limited capability of this family requires the functions themselves to be designed in a specific manner.

Component leveling:



FMT_LIM.1	Limited capabilities requires that the TSF is built to provide only the capabilities (perform action, gather information) necessary for its genuine purpose.
FMT_LIM.2	Limited availability requires that the TSF restrict the use of functions (refer to Limited capabilities (FMT_LIM.1)). This can be achieved, for instance, by removing or by disabling functions in a specific phase of the TOE’s life-cycle.
Management:	FMT_LIM.1, FMT_LIM.2 There are no management activities foreseen.
Audit:	FMT_LIM.1, FMT_LIM.2 There are no actions defined to be auditable.

The TOE Functional Requirement “Limited capabilities (FMT_LIM.1)” is specified as follows.

FMT_LIM.1 Limited capabilities

Hierarchical to:	No other components
Dependencies:	FMT_LIM.2 Limited availability.

FMT_LIM.1.1	The TSF shall be designed in a manner that limits their capabilities so that in conjunction with “Limited availability (FMT_LIM.2)” the following policy is enforced [assignment: *Limited capability and availability policy*].

The TOE Functional Requirement “Limited availability (FMT_LIM.2)” is specified as follows.

FMT_LIM.2 Limited availability

Hierarchical to:	No other components
Dependencies:	FMT_LIM.1 Limited capabilities.

FMT_LIM.2.1	The TSF shall be designed in a manner that limits their availability so that in conjunction with “Limited capabilities (FMT_LIM.1)” the following policy is enforced [assignment: *Limited capability and availability policy*].

Application note: The functional requirements FMT_LIM.1 and FMT_LIM.2 assume that there are two types of mechanisms (limited capabilities and limited availability) which together shall provide protection in order to enforce the policy. This also allows that

(i) the TSF is provided without restrictions in the product in its user environment but its capabilities are so limited that the policy is enforced

or conversely

(ii) the TSF is designed with test and support functionality that is removed from, or disabled in, the product prior to the Operational Use Phase.

The combination of both requirements shall enforce the policy.

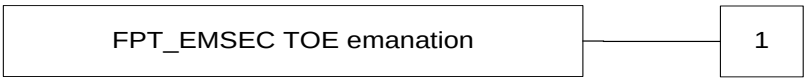
7.1.3 Definition of the Family FPT_EMS

The sensitive family FPT_EMS (TOE Emanation) of the Class FPT (Protection of the TSF) is defined here to describe the IT security functional requirements of the TOE. The TOE shall prevent attacks against the TOE and other secret data where the attack is based on external observable physical phenomena of the TOE. Examples of such attacks are evaluation of TOE's electromagnetic radiation, simple power analysis (SPA), differential power analysis (DPA), timing attacks, etc. This family describes the functional requirements for the limitation of intelligible emanations which are not directly addressed by any other component of CC part 2 [CC-2].

The family "TOE Emanation (FPT_EMS)" is specified as follows.

Family behaviour
This family defines requirements to mitigate intelligible emanations.

Component levelling:



FPT_EMS.1 TOE emanation has two constituents:

FPT_EMS.1.1 Limit of Emissions requires to not emit intelligible emissions enabling access to TSF data or user data.

FPT_EMS.1.2 Interface Emanation requires to not emit interface emanation enabling access to TSF data or user data.

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

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

FPT_EMS.1 TOE Emanation

Hierarchical to: No other components
Dependencies: No dependencies.

FPT_EMS.1.1 The TOE shall not emit [assignment: *types of emissions*] in excess of [assignment: *specified limits*] enabling access to [assignment: *list of types of TSF data*] and [assignment: *list of types of user data*].

FPT_EMS.1.2 The TSF shall ensure [assignment: *type of users*] are unable to use the following interface [assignment: *type of connection*] to gain access to [assignment: *list of types of TSF data*] and [assignment: *list of types of user data*].

7.1.4 Definition of the Family FIA_API

To describe the IT security functional requirements of the TOE, the family FIA_API of the class FIA (Identification and authentication) is defined here. This family describes the functional requirements for proof of the claimed identity for the authentication verification by an external entity, where the other families of the class FIA address the verification of the identity of an external entity.

Application Note 9: Other families of the class FIA describe only the authentication verification of the user's identity performed by the TOE and do not describe the functionality of the TOE to prove its own identity. The following paragraph defines the family FIA_API in the style of Common Criteria part 2 (cf. [3], chapter 'Extended components definition (APE_ECD)') from a TOE point of view.

FIA_API Authentication Proof of Identity

Family behaviour

This family defines functions provided by the TOE to prove its identity and to be verified by an external entity in the TOE IT environment.

Component levelling:

FIA_API.1 Authentication Proof of Identity.

Management FIA_API.1

The following actions could be considered for the management functions in FMT: Management of authentication information used to prove the claimed identity.

Audit: FIA_API.1

There are no actions defined to be auditable.

FIA_API.1 Authentication Proof of Identity

Hierarchical to:

No other components

Dependencies:

No dependencies

FIA_API.1.1

The TSF shall provide a [assignment: authentication mechanism] to prove the identity of the [assignment: authorized user or role, or of the TOE itself].

7.2 SECURITY FUNCTIONAL REQUIREMENTS

7.2.1 JCS Protection profile

This section states the security functional requirements for the Java Card System – Open configuration. For readability and for compatibility with previous versions, requirements are arranged into groups. All the groups defined in the table below apply to this Protection Profile.

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 2.2 feature. This group is the union of requirements from the Core (CoreG) and the Logical channels (LCG) groups defined in [PP/0305]. (cf Java Card System Protection Profile Collection [PP JCS]).
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 2.2.
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 2.2 feature.
Secure carrier (CarG)	The CarG group contains minimal requirements for secure downloading of applications on the card. This group contains the security requirements for preventing, the installation of a package that has not been bytecode verified, or that has been modified after bytecode verification.

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.

Subjects (prefixed with an "S") are described in the following table:

Subject	Description
S.ADEL	The applet deletion manager which also acts on behalf of the card issuer. It may be an applet ([JCRE22], §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 §7.2.1.3.
S.APPLET	Any applet instance.
S.BCV	The bytecode verifier (BCV), which acts on behalf of the verification authority who is in charge of the bytecode verification of the packages. This subject is involved in the PACKAGE LOADING security policy defined in §7.2.2.
S.CAD	The CAD represents off-card entity that communicates with the S.INSTALLER. The TOE does not provide JCRMI functionality.
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.

Subject	Description
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.

Objects (prefixed with an "O") are described in the following table:

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.

Information (prefixed with an "I") is described in the following table:

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 (used in enforcing the SFRs):

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
Context	Package AID, or "Java Card RE"
Currently Active Context	Package AID, or "Java Card RE"
Dependent package AID	Allows the retrieval of the Package AID and Applet's version number ([JCVM22], §4.5.2).
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 instance registered on the card
ResidentPackages	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.

Operations (prefixed with "OP") are described in the following table. Each operation has a specific number of 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.

Operation	Description
OP.ARRAY_ACCESS(O.JAVAOBJECT, field)	Read/Write an array component.
OP.ARRAY_LENGTH (O.JAVAOBJECT, field)	Get length of an array component.
OP.ARRAY_AASTORE(O.JAVAOBJECT, field)	Store into reference 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.
OP.DELETE_PCKG(O.CODE_PKG,...)	Delete a package, either logically or physically.
OP.DELETE_PCKG_APPLET(O.CODE_PKG,...)	Delete a package and its installed applets, 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, OP.TYPE_ACCESS. OP.ARRAY_LENGTH
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 instance of on an object in order to access to classes (standard or shareable interfaces objects).

(*) 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 JavaCardClass attribute's value is chosen by the creator.

7.2.1.1 CoreG_LC

This group is focused on the main security policy of the Java Card System, known as the firewall.

7.2.1.1.1 Firewall Policy

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
- OP.ARRAY_LENGTH
- OP.ARRAY_AASTORE**.**

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.

Application 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.

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:

Subject/Object	Attributes
S.PACKAGE	**LC Applet 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) An S.PACKAGE may freely perform any of 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) An S.PACKAGE may freely perform any of 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) An 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,) An 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 applies:**

  (a) The value of the attribute Selection Status of the package whose AID is "Package AID" is "Multiselectable»,

  (b) 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 ActiveApplets.

- **R.JAVA.5 S.PACKAGE may perform OP.CREATE upon O.JAVAOBJECT only if the value of the Sharing parameter(*) is "Standard" or "SIO".**

- **R.JAVA.6 ([JCRE3], §6.2.8): S.PACKAGE may freely perform OP.ARRAY_ACCESS or OP.ARRAY_LENGTH upon any O.JAVAOBJECT whose Sharing attribute has value "global array".**

FDP_ACF.1.3/FIREWALL The TSF shall explicitly authorize 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**.
3) **S.PACKAGE performing OP.ARRAY_AASTORE of the reference of an O.JAVAOBJECT whose sharing attribute has value "global array" or "Temporary JCRE entry point".**
4) **S.PACKAGE performing OP.PUTFIELD or OP.PUTSTATIC of the reference of an O.JAVAOBJECT whose sharing attribute has value "global array" or "Temporary JCRE entry point"**

Application Note: **FDP_ACF.1.4/FIREWALL**:

- The deletion of applets may render some O.JAVAOBJECT inaccessible, and the Java Card RE may be in charge of this aspect. This can be done, for instance, by ensuring that references to objects belonging to a deleted application are considered as a null reference. Such a mechanism is implementation-dependent.

In the case of an array type, fields are components of the array ([JVM], §2.14, §2.7.7), as well as the length; the only methods of an array object are those inherited from the Object class.

The Sharing attribute defines four categories of objects:

- Standard ones, whose both fields and methods are under the firewall policy,
- Shareable interface Objects (SIO), which provide a secure mechanism for inter-applet communication,
- JCRE entry points (Temporary or Permanent), who have freely accessible methods but protected fields,
- Global arrays, having both unprotected fields (including components; refer to JavaCardClass discussion above) and methods.

When a new object is created, it is associated with the Currently Active Context. But the object is owned by the applet instance within the Currently Active Context when the object is instantiated ([JCRE3], §6.1.3). An object is owned by an applet instance, by the JCRE or by the package library where it has been defined (these latter objects can only be arrays that initialize static fields of packages).

([JCRE3], Glossary) Selected Applet Context. The Java Card RE keeps track of the currently selected Java Card applet. Upon receiving a SELECT command with this applet's AID, the Java Card RE makes this applet the Selected Applet Context. The Java Card RE sends all APDU commands to the Selected Applet Context.

While the expression "Selected Applet Context" refers to a specific installed applet, the relevant aspect to the policy is the context (package AID) of the selected applet. In this policy, the "Selected Applet Context" is the AID of the selected package.

([JCRE3], §6.1.2.1) At any point in time, there is only one active context within the Java Card VM (this is called the Currently Active Context).

It should be noticed that the invocation of static methods (or access to a static field) is not considered by this policy, as there are no firewall rules. They have no effect on the active context as well and the "acting package" is not the one to which the static method belongs to in this case.
It should be noticed that the Java Card platform, version 2.2.x and version 3 Classic Edition, introduces the possibility for an applet instance to be selected on multiple logical channels at the same time, or accepting other applets belonging to the same package being selected simultaneously. These applets are referred to as multiselectable applets. Applets that belong to a same package are either all multiselectable or not ([JCVM3], §2.2.5). Therefore, the selection mode can be regarded as an attribute of packages. No selection mode is defined for a library package.

An applet instance will be considered an active applet instance if it is currently selected in at least one logical channel. An applet instance is the currently selected applet instance only if it is processing the current command. There can only be one currently selected applet instance at a given time ([JCRE3], §4).

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)**.

Application note:
References of temporary Java Card RE entry points, which cannot be stored in class variables, instance variables or array components, are transferred from the internal memory of the Java Card RE (TSF data) to some stack through specific APIs (Java Card RE owned exceptions) or Java Card RE invoked methods (such as the process (APDU apdu)); these are causes of OP.PUT (S1, S2, I) operations as well.

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:

Subject / Information	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 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 The TSF shall enforce **no additional information flow control SFP rules**.

FDP_IFF.1.4/JCVM The TSF shall explicitly authorize an information flow based on the following rules: **no additional information flow control SFP rules**.

FDP_IFF.1.5/JCVM The TSF shall explicitly deny an information flow based on the following rules: **no additional information flow control SFP rules**.

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**.

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**.

Application note:
The modification of the Selected Applet Context is performed in accordance with the rules given in [JCRE3], §4 and [JCVM3], §3.4.

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)**.

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

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**.

FMT_MSA.3/FIREWALL Static attribute initialization

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 The TSF shall allow **the [none]** to specify alternative initial values to override the default values when an object or information is created.

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 The TSF shall allow the **[none]** to specify alternative initial values to override the default values when an object or information is created.

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**

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.

7.2.1.1.2 Application Programming Interface

The following SFRs are related to the Java Card API.

The whole set of cryptographic algorithms is generally not implemented because of limited memory resources and/or limitations due to exportation. Therefore, the following requirements only apply to the implemented subset.

It should be noticed that the execution of the additional native code is not within the TSF. Nevertheless, access to API native methods from the Java Card System is controlled by TSF because there is no difference between native and interpreted methods in their interface or invocation mechanism.

FCS_CKM.1 Cryptographic key generation

FCS_CKM.1/RSA Cryptographic key generation

FCS_CKM.1.1/RSA The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **RSA Standard and RSA CRT Key Pair Generation** and specified cryptographic key sizes **512 to 3072 bits by steps of 32 bits** that meet the following: **see application note**.

Application note: the keys are generated and diversified in accordance with [JCAPI305] in classes KeyBuilder (buildKey method) and KeyPair (genKeyPair method).

FCS_CKM.1/ECDSA	Cryptographic key generation

FCS_CKM.1.1/ECDSA The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **ECDSA Key Pair Generation** and specified cryptographic key sizes **[P ranging from 160 to 521 bits]** that meet the following: **see application note**.

Application note: the keys are generated and diversified in accordance with [JCAPI305] in classes KeyBuilder (buildKey method) and KeyPair (genKeyPair method).

FCS_CKM.1/HMAC	Cryptographic key generation

FCS_CKM.1.1/HMAC The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **HMAC Key generation** and specified cryptographic key sizes **[see application note]** that meet the following: **[JCAPI305] standard**.

Application note

In accordance with [JCAPI305], the keys are generated and diversified in class KeyBuilder (buildKey method); the related key class is HMACKey of javacard.security.

As mentioned in [JCAPI305] the key can be of any length, but it is strongly recommended that the key is not shorter than the byte length of the hash output used in the HMAC implementation. Keys with length greater than the hash block length are first hashed with the hash algorithm used for the HMAC implementation. As required, the implementation also supports an HMAC key length equal to the length of the supported hash algorithm block size.

FCS_CKM.1/TDES	Cryptographic key generation

FCS_CKM.1.1/TDES The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **TDES Key generation** and specified cryptographic key sizes **112 bits for TDES 2 keys, 168 bits for TDES 3 keys** that meet the following: **[JCAPI305] standard**.

Application note: the keys are generated and diversified in accordance with [JCAPI305] in class KeyBuilder (buildKey method).

FCS_CKM.1/AES	Cryptographic key generation

FCS_CKM.1.1/AES The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **AES Key generation** and specified cryptographic key sizes 128, 192 and 256 **bits** that meet the following: **[JCAPI305] standard**.

Application note: the keys are generated and diversified in accordance with [JCAPI305] in class KeyBuilder (buildKey method).

FCS_CKM.1/ECPF	Cryptographic key generation

FCS_CKM.1.1/ECPF The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **ECC Key generation** and specified cryptographic key sizes 160, 192, 224, 256, 320, 384, 512, 521 **bits** that meet the following: **see application note.**

Application note: the keys are generated in accordance with [ANSI X9.62, FIPS PUB 186-4 standard].

FCS_CKM.1/ECDH	Cryptographic key generation

FCS_CKM.1.1/ECDH The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **EC Diffie-Hellman** and specified cryptographic key sizes 160, 192, 224, 256, 320, 384, 512, 521 **bits** that meet the following: **see application note.**

Application note: the keys are generated in accordance with [ANSI X9.63, FIPS PUB 186-4 standard].

FCS_CKM.1/DHGen Cryptographic key generation

FCS_CKM.1.1/DHGen The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **DH key generation** and specified cryptographic key sizes 1024, 1280, 1536, 2048 **bits** that meet the following: **see application note.**

Application note: the keys are generated in accordance with [ANSI X9.42, FIPS PUB 186-4 standard].

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 **[see application note]** that meets the following: **[JCAPI305] standard**.

Application note:

- The keys are reset as specified in [JCAPI305] Key class, with the method clearKey(). Any access to a cleared key for ciphering or signing shall throw an exception.

FCS_COP.1 Cryptographic operation

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

Iteration	Crypto operation	Crypto algorithm	Crypto Key size	List of Standards
/RSA-SIGN	signature & verification	RSA (STD) RSA (CRT)	512 to 3072 bits by steps of 32 bits	PKCS #1 Version 2.1, PKCS#1-PSS (IEEE 1363-2000), [ISO9796-2] RFC2409
/RSA-CIPHER	Encryption & decryption	RSA (STD) RSA (CRT)	512 to 3072 bits by steps of 32 bits	PKCS #1 Version 2.1, PKCS#1OAEP scheme (IEEE 1363-2000)
/TDES-CIPHER	Encryption & decryption	TDES	112 168	FIPS PUB 46-3, FIPS PUB 81, [ISO9797-1], PKCS#5
/TDES-MAC	Signature, Verification	TDES	112 168	FIPS PUB 46-3, FIPS PUB 81, [ISO9797-1], PKCS#5
/AES-CIPHER	Encryption & decryption	AES	128, 192, 256	[FIPS PUB 197], [NIST-SP800-38A], [ISO9797-1]
/AES-MAC	Signature, Verification	AES	128, 192, 256	[FIPS PUB 197], [NIST-SP800-38A]
/Hash	Hashing	see application note	None	[FIPS180-4]

/ECDSA_SIGN	Signature, Verification	ECDSA	P ranging from 160 to 521 bits	FIPS PUB 186-2
/ECDH	Secret Key Agreement	Elliptic Curve Diffie-Hellman (ECDH)	P ranging from 160 to 521 bits	IEEE P1363
/HMAC	Computation of a HMAC value	HMAC with hash algorithms mentioned in the application note below	see application note	rfc2104 FIPS PUB 198-1
/ECDSA_KEY_GEN	Key Pair Generation	ECDSA	P ranging from 160 to 521 bits	[FIPS PUB 186-4]
/DH_KEY_GEN	Key Pair Generation	DSA	512 to 2048 bits by steps of 32 bits	[FIPS PUB 186-4]

Application notes:

- The following TDES ciphers from [JCAPI305] are implemented:

Mode	Padding scheme	Field name in [JCAPI301] Cipher class
CBC	None (no padding)	ALG_DES_CBC_NOPAD
CBC	ISO9797 method 1	ALG_DES_CBC_ISO9797_M1
CBC	ISO9797 method 2	ALG_DES_CBC_ISO9797_M2
CBC	PKCS#5	ALG_DES_CBC_PKCS5
ECB	None (no padding)	ALG_DES_ECB_NOPAD
ECB	ISO9797 method 1	ALG_DES_ECB_ISO9797_M1
ECB	ISO9797 method 2	ALG_DES_ECB_ISO9797_M2
ECB	PKCS#5	ALG_DES_ECB_PKCS5

- The following TDES MACs from [JCAPI305] are implemented:

MAC length	MAC algorithm	Padding scheme	Field name in [JCAPI301] Signature class
4 bytes	ISO9797-1 MAC algorithm 3	ISO9797-1 method 2	ALG_DES_MAC4_ISO9797_1_M2_ALG3
4 bytes	3DES in outer CBC mode	ISO9797-1 method 1	ALG_DES_MAC4_ISO9797_M1
4 bytes	3DES in outer CBC mode	ISO9797-1 method 2	ALG_DES_MAC4_ISO9797_M2
4 bytes	3DES in outer CBC mode	PKCS#5	ALG_DES_MAC4_PKCS5
4 bytes	3DES in outer CBC mode	None	ALG_DES_MAC4_NOPAD
8 bytes	ISO9797-1 MAC algorithm 3	ISO9797-1 method 2	ALG_DES_MAC8_ISO9797_1_M2_ALG3
8 bytes	3DES in outer CBC mode	ISO9797-1 method 1	ALG_DES_MAC8_ISO9797_M1
8 bytes	3DES in outer CBC mode	ISO9797-1 method 2	ALG_DES_MAC8_ISO9797_M2
8 bytes	3DES in outer CBC mode	PKCS#5	ALG_DES_MAC8_PKCS5
8 bytes	3DES in outer CBC mode	None	ALG_DES_MAC8_NOPAD

- The following AES ciphers from [JCAPI305] are implemented:

Mode	Padding scheme	Field name in [JCAPI301] Cipher class
CBC	None (no padding)	ALG_AES_BLOCK_128_CBC_NOPAD
CBC	ISO9797 method 1	ALG_AES_CBC_ISO9797_M1
CBC	ISO9797 method 2	ALG_AES_CBC_ISO9797_M2
CBC	PKCS#5	ALG_AES_CBC_PKCS5
ECB	None (no padding)	ALG_AES_BLOCK_128_ECB_NOPAD
ECB	ISO9797 method 1	ALG_AES_ECB_ISO9797_M1
ECB	ISO9797 method 2	ALG_AES_ECB_ISO9797_M2
ECB	PKCS#5	ALG_AES_ECB_PKCS5

- The following AES MACs from [JCAPI305] are implemented:

MAC length	MAC algorithm	Padding scheme	Field name in [JCAPI301] Signature class
16 bytes	AES in CBC mode, block size 128 bits	None	ALG_AES_MAC_128_NOPAD
24 bytes	AES in CBC mode, block size 192 bits	None	ALG_AES_MAC_192_NOPAD
32 bytes	AES in CBC mode, block size 256 bits	None	ALG_AES_MAC_256_NOPAD

- The following RSA signatures from [JCAPI305] are implemented:

Hash algorithm	Padding scheme	Field name in [JCAPI301] Signature class
SHA224	PKCS#1	ALG_RSA_SHA_224_PKCS1
SHA224	PKCS#1-PSS scheme (IEEE 1363-2000)	ALG_RSA_SHA_224_PKCS1_PSS
SHA256	PKCS#1	ALG_RSA_SHA_256_PKCS1
SHA256	PKCS#1-PSS scheme (IEEE 1363-2000)	ALG_RSA_SHA_256_PKCS1_PSS
SHA384	PKCS#1	ALG_RSA_SHA_384_PKCS1
SHA384	PKCS#1-PSS scheme (IEEE 1363-2000)	ALG_RSA_SHA_384_PKCS1_PSS
SHA512	PKCS#1	ALG_RSA_SHA_512_PKCS1
SHA512	PKCS#1-PSS scheme (IEEE 1363-2000)	ALG_RSA_SHA_512_PKCS1_PSS
SHA1	ISO 9796-2	ALG_RSA_SHA_ISO9796
SHA1	PKCS#1	ALG_RSA_SHA_PKCS1
SHA1	PKCS#1-PSS scheme (IEEE 1363-2000)	ALG_RSA_SHA_PKCS1_PSS
SHA1	RFC2409	ALG_RSA_SHA_RFC2409

- The following RSA ciphers from [JCAPI305] are implemented:

[JCAPI301] class	Implemented algorithms
Cipher	ALG_RSA_NOPAD
	ALG_RSA_PKCS1
	ALG_RSA_PKCS1_OAEP

- The following ECDSA signatures from [JCAPI305] are implemented:

Hash algorithm	Field name in [JCAPI301] Signature class
SHA1	ALG_ECDSA_SHA
SHA224	ALG_ECDSA_SHA_224
SHA256	ALG_ECDSA_SHA_256
SHA384	ALG_ECDSA_SHA_384
SHA512	ALG_ECDSA_SHA_512

- ECDH: The secret keys are derived using the KeyAgreement class (generateSecret method) of javacard.security.

- The following hash algorithms from [JCAPI305] are implemented:

Hash algorithm	Field name in [JCAPI301] MessageDigest class
SHA1	ALG_SHA
SHA224	ALG_SHA_224
SHA256	ALG_SHA_256
SHA384	ALG_SHA_384
SHA512	ALG_SHA_512

- The following HMAC algorithms from [JCAPI305] are implemented:

Hash algorithm used in HMAC computation	Field name in [JCAPI301] Signature class
SHA1	ALG_HMAC_SHA1
SHA256	ALG_HMAC_SHA_256
SHA384	ALG_HMAC_SHA_384
SHA512	ALG_HMAC_SHA_512

As mentioned in [JCAPI305] the key can be of any length, but it is strongly recommended that the key is not shorter than the byte length of the hash output used in the HMAC implementation. Keys with length greater than the hash block length are first hashed with the hash algorithm used for the HMAC implementation. As required, the implementation also supports an HMAC key length equal to the length of the supported hash algorithm block size.

Random Numbers

The TOE generates random numbers. To define the IT security functional requirements of the TOE an additional family (FCS_RNG) of the Class FCS (cryptographic support) is defined in chapter 7.1. This family FCS_RNG Generation of random numbers describes the functional requirements for random number generation used for cryptographic purposes.

The TOE shall meet the requirement “Quality metric for random numbers (FCS_RNG.1)” as specified below (Common Criteria Part 2 extended).

FCS_RNG.1 Random number generation

FCS_RNG.1.1 The TSF shall provide a [**hybrid deterministic**] random number generator that implements: [**enhanced backward secrecy & enhanced forward secrecy**].

FCS_RNG.1.2 The TSF shall provide random numbers that pass [**[AIS31] test procedure A**].

Application notes:

- The random number generator is compliant with AIS DRG.4.
- The cryptographic algorithm is Hybrid Deterministic RNG based on AES-256

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**.

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**.

FDP_RIP.1/GlobalArray Subset residual information protection

FDP_RIP.1.1/GlobalArray [*Refined*] The TSF shall ensure that any previous information content of a resource is made unavailable upon **deallocation of the resource from** the applet as a result of returning from the process method to the following objects: **a user Global Array**.

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**.

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)**.

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**.

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.JAVAOBJECTs**.

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 [JCAPI3]**.

7.2.1.1.3 Card Security Management

FAU_ARP.1 Security alarms

FAU_ARP.1.1 The TSF shall take **the following actions:**
- **throw an exception,**
- **or lock the card session**
- **or reinitialize the Java Card System and its data**
- **[assignment: no other actions]**

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
- Card Manager life cycle inconsistency
- Random trap detection

FDP_SDI.2/DATA Stored data integrity monitoring and action

FDP_SDI.2.1/DATA The TSF shall monitor user data stored in containers controlled by the TSF for **integrity errors** on all objects, based on the following attributes: **integrity-sensitive data**.

FDP_SDI.2.2/DATA Upon detection of a data integrity error, the TSF shall **decrease the counter of fault detection and set the card in degraded mode if counter reaches value 0**.

Application note:
The following data persistently stored by TOE have the **integrity-sensitive data** attribute:
- Key (i.e. objects instance of classes implemented the interface Key)
- PIN (objects instance of class OwnerPin)
- Package.

FPR_UNO.1 Unobservability

FPR_UNO.1.1 The TSF shall ensure that **unauthorized users** are unable to observe the operation **cryptographic operations / comparisons operations** on **Key values / PIN values** by **S.JCRE, S.Applet**.

FPT_FLS.1/JCS Failure with preservation of secure state

FPT_FLS.1.1/JCS The TSF shall preserve a secure state when the following types of failures occur: **those associated to the potential security violations described in FAU_ARP.1**.

Application note:

The Java Card RE Context is the Current context when the Java Card VM begins running after a card reset ([JCRE3], §6.2.3) or after a proximity card (PICC) activation sequence ([JCRE3]). Behavior of the TOE on power loss and reset is described in [JCRE3], §3.6 and §7.1. Behavior of the TOE on RF signal loss is described in [JCRE3], §3.6.1.

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 argument,** 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**
- **The rules defined in ISO 7816-6**
- **The rules defined in [GP23] specification**

when interpreting the TSF data from another trusted IT product.

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

7.2.1.1.4 AID Management

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**
- **Applet's version number**
- **registered applet's AID**
- **applet selection status.**

Refinement:
- "Individual users" stands for applets.

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.

Application notes:
- By users here it must be understood the ones associated to the packages (or applets) that act as subjects of policies. In the Java Card System, every action is always performed by an identified user interpreted here as the currently selected applet or the package that is the subject's owner. Means of identification are provided during the loading procedure of the package and the registration of applet instances.
- The role Java Card RE defined in FMT_SMR.1 is attached to an IT security function rather than to a "user" of the CC terminology. The Java Card RE does not "identify" itself with respect to the TOE, but it is a part of it.

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: **Package AID are defined with associated value during loading and with context identifier**.

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: **[None]**

Application note:

- The user is the applet and the subject is the S.PACKAGE. The subject security attribute "Context" shall hold the user security attribute "package AID".

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**.

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**.

7.2.1.2 INSTG

This group combines 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 modeled as an 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).

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

FDP_ITC.2.1/Installer The TSF shall enforce the **PACKAGE LOADING information flow control SFP** when importing user data, controlled under the SFP, from outside of the TOE.

Application note:

- The most common importation of user data is package loading and applet installation on the behalf of the installer. Security attributes consist of the shareable flag of the class component, AID and version numbers of the package, maximal operand stack size and number of local variables for each method, and export and import components (accessibility).

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.

Application note:

- The format of the CAP file is precisely defined in Sun's specification ([JCVM3]); it contains the user data (like applet's code and data) and the security attribute altogether. Therefore there is no association to be carried out elsewhere.

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.

Application note:

- Each package contains a package Version attribute, which is a pair of major and minor version numbers ([JCVM3], §4.5). With the AID, it describes the package defined in the CAP file. When an export file is used during preparation of a CAP file, the versions numbers and AIDs indicated in the export file are recorded in the CAP files ([JCVM3], §4.5.2): the dependent packages Versions and AIDs attributes allow the retrieval of these identifications.. Implementation-dependent checks may occur on a case-by-case basis to indicate that package files are binary compatibles. However, package files do have "package Version Numbers" ([JCVM3]) used to indicate binary compatibility or incompatibility between successive implementations of a package, which obviously directly concern this requirement.

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).

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.

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**.

FPT_RCV.3/Installer Automated recovery without undue loss

FPT_RCV.3.1/Installer When automated recovery from **a failure or service discontinuity** is not possible, the TSF shall enter a maintenance mode where the ability to return to a secure state is provided.

Application notes:

This element is not within the scope of the Java Card specification, which only mandates the behavior of the Java Card System in good working order. Further details on the "maintenance mode" shall be provided in specific implementations. The following is an excerpt from [CC2], p296: In this maintenance mode normal operation might be impossible or severely restricted, as otherwise insecure situations might occur. Typically, only authorized users should be allowed access to this mode but the real details of who can access this mode is a function of FMT: Security management. If FMT: Security management does not put any controls on who can access this mode, then it may be acceptable to allow any user to restore the system if the TOE enters such a state. However, in practice, this is probably not desirable as the user restoring the system has an opportunity to configure the TOE in such a way as to violate the SFRs.

FPT_RCV.3.2/Installer For **[detection of a potential loss of integrity during the transmission of an Executable Load File to the card, abortion of the installation process of an Executable Load File, or any fatal error occurred during the linking of an Executable Load File to the Executable Files already**

installed on the card], the TSF shall ensure the return of the TOE to a secure state using automated procedures.

Application notes:

Should the installer fail during loading/installation of a package/applet, it has to revert to a "consistent and secure state". The Java Card RE has some clean up duties as well; see [JCRE3], §11.1.5 for possible scenarios. Precise behavior is left to implementers. This component shall include among the listed failures the deletion of a package/applet. See ([JCRE3], 11.3.4) for possible scenarios. Precise behavior is left to implementers.

Other events such as the unexpected tearing of the card, power loss, and so on, are partially handled by the underlying hardware platform (see [PP0084b]) and, from the TOE's side, by events "that clear transient objects" and transactional features. See FPT_FLS.1.1/JCS, FDP_RIP.1/TRANSIENT, FDP_RIP.1/ABORT and FDP_ROL.1/FIREWALL.

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 **[the loss of the Executable Load File being installed]** for loss of TSF data or objects under the control of the TSF.

Application notes:

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

FPT_RCV.3.4/Installer The TSF shall provide the capability to determine the objects that were or were not capable of being recovered.

Application note:

Additional code, if any, will be also loaded and installed using such rules as a standard executable load file with specific attributes handled with GemActivate.

7.2.1.3 ADELG

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 phase and therefore requires specific treatment.

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.

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:

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**
O.JAVAOBJECT	**Owner, Remote**

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 if one of the following conditions holds:

(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], §11.3.4.2, 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 ([JCRE3], §8.5) O.JAVAOBJECT is remote reachable.

- **R.JAVA.15 ([JCRE3], §11.3.4.2.1, 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 in the context of O.APPLET 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], §11.3.4.3, Applet/Library Package Deletion). The S.ADEL may perform OP.DELETE_PCKG upon an O.CODE_PCKG only if,**
  (1) S.ADEL is currently selected,
  (2) no reachable O.JAVAOBJECT, from a package distinct from O.CODE_PCKG that is an instance of a class that belongs to O.CODE_PCKG exists on the card and
  (3) there is no resident package on the card that depends on O.CODE_PCKG.

- **R.JAVA.17 ([JCRE3], §11.3.4.4, Applet Package and Contained Instances Deletion). S.ADEL may perform OP.DELETE_PCKG_APPLET upon an O.CODE_PCKG only if,**
  (1) S.ADEL is currently selected,
  (2) no reachable O.JAVAOBJECT, from a package distinct from O.CODE_PCKG, which is an instance of a class that belongs to O.CODE_PCKG exists on the card,
  (3) there is no package loaded on the card that depends on O.CODE_PCKG 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 authorize access of subjects to objects based on the following additional rules: **none**.

FDP_ACF.1.4/ADEL The TSF shall explicitly deny access of subjects to objects based on the following additional rules:
any subject but S.ADEL to O.CODE_PKG or O.APPLET for the purpose of deleting them from the card.

Application notes:
FDP_ACF.1.2/ADEL:

- This policy introduces the notion of reachability, which provides a general means to describe objects that are referenced from a certain applet instance or package.
- S.ADEL calls the "uninstall" method of the applet instance to be deleted, if implemented by the applet, to inform it of the deletion request. The order in which these calls and the dependencies checks are performed are out of the scope of this security target.

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**.

Application note:

- Requirements on de-allocation during applet/package deletion are described in [JCRE3], §11.3.4.2, §11.3.4.3 and §11.3.4.4.

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 (S.JCRE)**.

FMT_MSA.3/ADEL Static attribute initialization

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.

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**.

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.

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**.

Application note:

- The applet instance deletion must be atomic. The "secure state" referred to in the requirement must comply with the Java Card specifications ([JCRE3], §11.3.4). That is, if a reset or power fail occurs during the deletion process, then before any applet is selected in card, either the applet instance deletion is completed or the applet shall be selectable and all objects owned by the applet remain unchanged (that is, the functionality of all applet instances on the card remains the same as prior to the unsuccessful deletion attempt) [JCRE3], §11.3.4.

7.2.1.4 ODELG

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.

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().

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**.

7.2.1.5 CarG

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

FCO_NRO.2/CM Enforced proof of origin

FCO_NRO.2.1/CM The TSF shall enforce the generation of evidence of origin for transmitted **application packages** at all times.

Application note:
Upon reception of a new application package for installation, the card manager shall first check that it actually comes from the verification authority. The verification authority is the entity responsible for bytecode verification.

FCO_NRO.2.2/CM [Editorially Refined] The TSF shall be able to relate the **identity** of the originator of the information, and the **application package contained in** the information to which the evidence applies.

FCO_NRO.2.3/CM The TSF shall provide a capability to verify the evidence of origin of information to **recipient** given **no limitation**.

FDP_IFC.2/CM Complete information flow control

FDP_IFC.2.1/CM The TSF shall enforce the **PACKAGE LOADING information flow control SFP** on **S.INSTALLER, S.BCV, S.CAD, and I.APDU** and all operations that cause that information to flow to and from subjects covered by the SFP.

FDP_IFC.2.2/CM 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.

Application note:

The subjects covered by this policy are those involved in the loading of an application package by the card through a potentially unsafe communication channel.

The operations that make information to flow between the subjects are those enabling to send a message through and to receive a message from the communication channel linking the card to the outside world. It is assumed that any message sent through the channel as clear text can be read by the attacker. Moreover, the attacker may capture any message sent through the communication channel and send its own messages to the other subjects.

The information controlled by the policy is the APDUs exchanged by the subjects through the communication channel linking the card and the CAD. Each of those messages contain part of an application package that is required to be loaded on the card, as well as any control information used by the subjects in the communication protocol.

FDP_IFF.1/CM Simple security attributes

FDP_IFF.1.1/CM The TSF shall enforce the **PACKAGE LOADING information flow control SFP** based on the following types of subject and information security attributes: **[the Command Security Level defined for the messages that the card receives through the secure channel]**.

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

FDP_IFF.1.3/CM The TSF shall enforce the **[possible security levels are: NO-SEC (clear text), C-AUTHENTICATED (authentication of the command's emitter), C-MAC (authentication of the emitter and integrity of the command), C-DEC (authentication of the emitter, integrity and confidentiality of the command)]**.

FDP_IFF.1.4/CM The TSF shall explicitly authorize an information flow based on the following rules: **[the SD may process:**

- **an (INITIALIZE-UPDATE) operation only if the key set specified in the command exist,**
- **an (EXTERNAL-AUTHENTICATE) operation if the following conditions are fulfilled:**
  1) **The cryptogram received from the off-card subject is equal to the cryptogram computed by the Security Domain.**
  2) **The MAC attached to the message has been generated using the CMAC session key and the current value of the ICV.**
- **a (GET-DATA) operation if the following condition are fulfilled:**
  1) **If the command security level is at least C-MAC,**
  2) **The MAC attached to the message has been generated from the command using the C-MAC session key and the current value of the ICV.**
- **any received operation for any other command if the following conditions hold:**
  1) **The current security level is at least AUTHENTICATED.**
  2) **If the command security level is at least C-MAC, the MAC attached to the message has been generated from the clear-text command using the C-MAC session key and the current value of the ICV.**

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

- **The TOE fails to verify the integrity and authenticity evidences of the application package**
- **[A Security Domain may always process a (SELECT) operation or a (Get DATA) operation at the security level NO-SEC].**

FDP_UIT.1/CM Data exchange integrity

FDP_UIT.1.1/CM The TSF shall enforce the **PACKAGE LOADING information flow control SFP** to be able to **receive** user data in a manner protected from **modification, deletion, insertion, and replay** errors.

FDP_UIT.1.2/CM [Refined] The TSF shall be able to determine on receipt of user data, whether **modification, deletion, insertion, replay of some of the pieces of the application sent by the CAD** has occurred.

Application note:

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

FIA_UID.1/CM Timing of identification

FIA_UID.1.1/CM The TSF shall allow **selection of a security domain and execution of Card Manager** on behalf of the user to be performed before the user is identified.

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

FMT_MSA.1/CM Management of security attributes

FMT_MSA.1.1/CM The TSF shall enforce the **PACKAGE LOADING information flow control SFP** to restrict the ability to **modify** the security attributes [Card Life Cycle, Security Level] to [Card Issuer for Card Manager, The Application Provider for APSD].

FMT_MSA.3/CM Static attribute initialization

FMT_MSA.3.1/CM The TSF shall enforce the **PACKAGE LOADING information flow control SFP** to provide **restrictive** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/CM The TSF shall allow **[None]** to specify alternative initial values to override the default values when an object or information is created.

FMT_SMF.1/CM Specification of Management Functions

FMT_SMF.1.1/CM The TSF shall be capable of performing the following management functions:

- **Modification of the Card life cycle inducing availability of management functions**.

FMT_SMR.1/CM Security roles

FMT_SMR.1.1/CM The TSF shall maintain the roles **[S.CAD, S.CARDMANAGER]**.

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

FTP_ITC.1/CM Inter-TSF trusted channel

FTP_ITC.1.1/CM The TSF shall provide a communication channel between itself and another trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure.

FTP_ITC.1.2/CM [Refined] The TSF shall permit **the CAD placed in the card issuer secured environment** to initiate communication via the trusted channel.

FTP_ITC.1.3/CM The TSF shall initiate communication via the trusted channel for **loading/ installing a new application package on the card**.

Application note:

- There is no dynamic package loading on the Java Card platform. New packages can be installed on the card only on demand of the card issuer.

7.2.2 Supplementary Security Functional Requirements

7.2.2.1 Smart Card Platform Security Functional Requirements

This group contains the security requirements for the smart card platform, that is, operating system and chip that the Java Card System is implemented upon. The requirements are expressed in terms of security functional requirements from [CC-2].

FPT_TST.1/SCP TSF Testing

FPT_TST.1.1/SCP The TSF shall run a suite of self-tests **periodically during normal operation** to demonstrate the correct operation of **security mechanisms of the IC**.

FPT_TST.1.2/SCP The TSF shall provide authorized users with the capability to verify the integrity of **Keys**.

FPT_TST.1.3/SCP The TSF shall provide authorized users with the capability to verify the integrity of **Applets, user PIN, user Keys**.

FPT_PHP.3/SCP Resistance to physical attacks

FPT_PHP.3.1/SCP The TSF shall resist [**physical manipulation and physical probing**] to the [**all TOE components implementing the TSF**] by responding automatically such that the SFRs are always enforced.

FPT_RCV.3/SCP Automated recovery without undue loss

FPT_RCV.3.1/SCP When automated recovery from **security policy violation** is not possible, the TSF shall enter a maintenance mode where the ability to return to a secure state is provided.

FPT_RCV.3.2/SCP For **execution access to a memory zone reserved for TSF data, writing access to a memory zone reserved for TSF's code, and any segmentation fault performed by a Java Card applet**, the TSF shall ensure the return of the TOE to a secure state using automated procedures.

FPT_RCV.3.3/SCP The functions provided by the TSF to recover from failure or service discontinuity shall ensure that the secure initial state is restored without exceeding **o the contents of Java Card static fields, instance fields, and array positions that fall under the scope of an open transaction; o the Java Card objects that were allocated into the scope of an open transaction; o the contents of Java Card transient objects; o any possible Executable Load File being loaded when the failure occurred** for loss of TSF data or objects under the control of the TSF.

FPT_RCV.3.4/SCP The TSF shall provide the capability to determine the objects that were or were not capable of being recovered.

FPT_RCV.4/SCP Function recovery

FPT_RCV.4.1/SCP The TSF shall ensure that **reading from and writing to static and objects' fields interrupted by power loss** have the property that the SF either completes successfully, or for the indicated failure scenarios, recovers to a consistent and secure state.

7.2.2.2 CMGR

The security requirements below help to define a policy for controlling access to card content management operations and for expressing card issuer security concerns.
Most of them come from [PP-JCS] but are instantiated to add more precisions regarding the TOE card content management

FDP_UIT.1/CCM Data exchange integrity

FDP_UIT.1.1/CCM The TSF shall enforce the **Secure Channel Protocol information flow control policy and the Security Domain access control policy** to **transmit and receive** user data in a manner protected from **modification, deletion, insertion and replay** errors.

FDP_UIT.1.2/CCM The TSF shall be able to determine on receipt of user data, whether **modification, deletion, insertion and replay** has occurred.

FDP_ROL.1/CCM	**Basic rollback**

FDP_ROL.1.1/CCM The TSF shall enforce **Security Domain access control policy** to permit the rollback of the **installation operation** on the **executable files and application instances (see application note)**.

FDP_ROL.1.2/CCM The TSF shall permit operations to be rolled back within the **size of the available memory when the card content management operation starts**.

Application note: patch is imported using such rules and loaded as a standard executable load file with specific attributes handled with GemActivate.

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

FDP_ITC.2.1/CCM The TSF shall enforce the **Security Domain access control policy and the Secure Channel Protocol information flow policy** 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: **The loading of a new Executable Load File is allowed only if, AID attribute of each dependent Executable File is equal to the identified AID in the CAP File, such AID is unique, SD is personalized and authorized to load. Otherwise, the load of ELF is rejected**.

Application notes:

- This Functional Component Instance enforces a security information flow control policy. Rules must be defined for importation operations. These rules must take into account all user data.
- Patch is also imported using such rules and loaded as a standard executable load file with specific attributes handled with GemActivate.

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 [GP23] §9.3.5**.

Application note: Patch is loaded as a standard executable load file with specific attributes handled with GemActivate.

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** in accordance with a specified cryptographic algorithm

- **PKC Scheme: SHA-1 hash and PKCS#1 RSA signature**
- **or DES Scheme: Single DES plus final Triple DES MAC (Retail MAC)**

and cryptographic key sizes

- **PKC Scheme: RSA key of minimum length 1024 bits**
- **DES Scheme: DES key of minimum length 16 bytes**

that meet the following:

- **Sections C.1.2 and C.6 of [GP23]**
- **PKC Scheme: SSA-PKCS1-v1_5 as defined in PKCS#1**
- **DES Scheme: ISO 9797-1 as MAC Algorithm 3 with output transformation 3, without truncation, and with DES taking the place of the block cipher**

FDP_ACC.1/SD	**Subset access control**

FDP_ACC.1.1/SD The TSF shall enforce the **Security Domain access control policy** on:

- **Subjects: S.INSTALLER, S.ADEL, S.CAD (from [PP-JCS]) and S.SD**
- **Objects: Delegation Token, DAP Block and Load File**
- **Operations: GlobalPlatform's card content management APDU commands and API methods**.

FDP_ACF.1/SD	**Security attribute based access control**

FDP_ACF.1.1/SD The TSF shall enforce the **Security Domain access control policy** to objects based on the following:

- **Subjects:**
  - **S.INSTALLER, defined in [PP-JCS] and represented by the GlobalPlatform Environment (OPEN) on the card, the Card Life Cycle attributes (defined in Section 5.1.1 of [GP23]);**
  - **S.ADEL, also defined in [PP-JCS] 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 [GP23]), a life-cycle status (defined in Section 5.3.2 of [GP23]) and a Secure Communication Security level (defined in Section 10.6 of [GP23]);**
  - **S.CAD, defined in [PP-JCS], 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.**
- **The following security attributes:**
  - **The Default Selected attribute specifies whether the applet instance is the one that should be executed when no application has been explicitly selected.**
  - **The Application State attribute specifies the current life cycle state of the application instance, which may be either SELECTABLE, APPLICATION_SPECIFIC, LOCKED.**
  - **The Card State attribute, is the current state in the life cycle of the card, which may be either OP_READY, INITIALIZED, SECURED, CARD_LOCKED, or TERMINATED.**
  - **The Card Lock attribute specifies whether the applet is allowed to temporary lock the services of the smart card.**
  - **The Card Termination attribute specifies whether the applet is allowed to definitely disable the services of the smart card.**
  - **The CVM attribute specifies whether the applet is allowed to modify the try limit and the PIN code of the global CVM service.**
  - **The Registered Applications attribute specifies the Executable Files and application instances that have been installed on the card so far and their dependencies**.

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: **Runtime behavior rules defined by GlobalPlatform for:**

- **loading (Section 9.3.5 of [GP23]);**
- **installation (Section 9.3.6 of [GP23]);**
- **extradition (Section 9.4.1 of [GP23]);**
- **registry update (Section 9.4.2 of [GP23]);**
- **content removal (Section 9.5 of [GP23])**.

FDP_ACF.1.3/SD The TSF shall explicitly authorize access of subjects to objects based on the following additional rules:

- **Rule SD-1: A card administration request may be accepted only if the APDU command specifying the request is well-formed according to [GP23].**
- **Rule SD-2: A card administration request other than requesting card management data may be accepted only if the Card State is not TERMINATED.**
- **Rule SD-3: The selection of an applet instance may be accepted only if the Applet State is not LOCKED.**
- **Rule SD-4: The update of the life cycle state of an application instance is accepted only if the new state is consistent with its current life cycle state according to GlobalPlatform's life cycle rules (either coming from an APDU command or from an application instance through the GP API).**
- **Rule SD-5: A request for installing an Executable Load File may be accepted only if there is enough resources for loading the Executable File, and no Executable File on the card has been already registered with the specified AID.**
- **Rule SD-6: A Executable Load File block may be loaded only if all its previous blocks have been received in order, and there are sufficient resources for storing the new one.**
- **Rule SD-7: A new applet instance may be created only if the Package Properties enables applet instantiation or multiple applet instances (if there is already an instance for that applet) but also if the AID specified for the applet instance is not already used for another applet or Executable File installed on the card, and the privileges specified for it are consistent with the GlobalPlatform rules specified in [GP23].**
- **Rule SD-8: An Executable File may be deleted from the smart card only if it is not reachable from other Executable Files or application instances on the card.**
- **Rule SD-9: An applet instance may be deleted from the card only it is not currently active on a logical channel, and none of the resources it has allocated is reachable from other Executable Files or Application instances installed on the card.**
- **Rule SD-10: An applet instance may lock the card only if it has the Card Lock privilege.**
- **Rule SD-11: An applet instance may terminate the card only if it has the Card Termination privilege.**
- **Rule SD-12: An applet instance may unlock the CVM service or modify the CVM try limit or PIN code only if it has the CVM privilege.**
- **Rule SD-13: A request involving the use of any of the Security domain keys is accepted only if the concerned keys are integer**.

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**.

FMT_MSA.1/SD	**Management of security attributes**

FMT_MSA.1.1/SD The TSF shall enforce the **Security Domain access control policy** to restrict the ability to **modify** the security attributes **Any security attributes registered the GP Registry such as:**

- **Application state of an application instance (1)**
- **Default selected application (2)**
- **Card Life cycle state (3)**
- **Package properties (4)**
- **Application association (5)**

to

- **the Security Domain and the application instance itself (1)**
- **the Security Domain (2&4)**
- **the Security Domain and application with privilege (Card Lock or Terminated) (3)**.

FMT_MSA.3/SD	Static attribute initialization

FMT_MSA.3.1/SD The TSF shall enforce the **Security Domain access control policy** to provide **restrictive** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/SD The TSF shall allow the **Issuer or authorized application provider** 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.
- The Default Selected application shall be the ISD.
- The initial value of the Application State of an applet instance shall be SELECTABLE.

Application notes:

When the TOE enters the life cycle phases under the scope of this Security Target, the Card State shall be at least SECURED. The initial value of the Application State of an applet instance shall be SELECTABLE. The initial Package Properties shall enable all card content management operations on the package.

The Issuer or authorized application provider may assign the Default Select privilege to another application instance.

FMT_SMF.1/SD	Specification of Management Functions

FMT_SMF.1.1/SD The TSF shall be capable of performing the following management functions:

- **Restricting the properties associated to a given package**
- **Registering a new Executable File or application instance in the GP registry.**
- **Removing the specified entries from the GP registry when a DELETE command is received.**
- **Unsetting it as the Default Select application and set this privilege to a new application instance.**
- **Granting the privileges that the authorized entities (OEM, or Application Provider) specifies when a new application instance is installed**.

FMT_SMR.1/SD	Security roles

FMT_SMR.1.1/SD The TSF shall maintain the roles

- **Issuer Security Domain**
- **Supplementary Security Domain**
- **Certification Authority Security Domain**.
- **Verification Authority Security Domain**

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

The section below states the security functional requirements for the Secure Channel.

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.

FTP_ITC.1.2/SC The TSF shall permit **another trusted IT product** 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:**

- **loading or deleting an Executable Load file;**
- **installing or removing an application instance;**
- **extrading an Executable Load file or an application instance;**
- **registry update;**
- **Loading or removing a Key Set;**
- **SD personalization;**
- **Changing the Application Life Cycle or card Life Cycle.**

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** at all times.

FCO_NRO.2.2/SC The TSF shall be able to relate the **identity** of the originator of the information, and the **Executable Load Files** 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 **originator** given **Executable load files**.

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** on

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

The subjects covered by this policy are those involved in the exchange of messages between the card and the CAD through a potentially unsafe communication channel:

- **An off-card subject that represents the authorized entities (S.BCV).**
- **Any application with the Security Domain privilege (S.CRD).**

The information controlled by this policy is the one contained in the APDU commands sent to the card and their associated responses returned to the CAD or the mobile.

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.

FDP_IFF.1/SC	**Simple security attributes**

FDP_IFF.1.1/SC The TSF shall enforce the **Secure Channel Protocol information flow control policy** 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 personalization.**

The subjects have the following security attributes for SCP02 [GP23]:

- **The Challenge is a random number generated by the subject in order to identify the current session.**
- **The Cryptogram is a secret relative to the current smart card session that serves to authenticate the on- and off-card subjects. The cryptogram is derived from the challenges of both the card and the terminal.**

- **The Key Set is a collection of three keys (Secure Channel Encryption Key (S-ENC), a Command Message Authentication Code Key (C-MAC) and a Data Encryption Key (DEK)) used to encrypt the Derivation Data in order to generate the session keys. It is identified by a key version number.**
- **The Session Keys is a set of keys derived from KeySet and sequence counter to be used to verify the origin and integrity of the received message, and to decrypt their contents. This set is made of the following keys:**
  - **Command Message Authentication Code Key (C-MAC session key);**
  - **Encryption Key (S-ENC session key);**
  - **Data Encryption Key (DEK session key).**
- **The Command Security Level defined for the messages that the card receives through the secure channel. The possible security levels are:**
  - **NO-SEC (clear text),**
  - **C-AUTHENTICATED (authentication of the command's issuer),**
  - **C-MAC (authentication of the issuer and integrity of the command),**
  - **C-DEC (authentication of the issuer, integrity and confidentiality of the command).**
- **The Initial Chaining Vector (ICV) is a value used to compute the MAC value of a message, which relates it to the previous messages of the current session.**

The subjects have the following security attributes for SCP03 [GP23 Amend D]:

- **The Challenge is a random number generated by the subject in order to identify the current session.**
- **The Cryptogram is a secret relative to the current smart card session that serves to authenticate the on- and off-card subjects. The cryptogram is derived from the challenges of both the card and the terminal.**
- **The Key Set is a collection of three keys (Static Secure Channel Encryption Key (Key-ENC), Static Secure Channel Message Authentication Code Key (Key-MAC) and a Data Encryption Key (Key-DEK)) used to encrypt the Derivation Data in order to generate the session keys. It is identified by a key version number.**
- **The Session Keys is a set of keys derived from KeySet and sequence counter to be used to verify the origin and integrity of the received message, and to decrypt their contents. This set is made of the following keys:**
  - **Secure Channel Message Authentication Code Key for Command (S-MAC);**
  - **Secure Channel Message Authentication Code Key for Response (S-RMAC);**
  - **Session Secure Channel Encryption Key (S-ENC).**
- **The Command Security Level defined for the messages that the card receives through the secure channel. The possible security levels are:**
  - **NO-SEC (clear text),**
  - **AUTHENTICATED (authentication of the command's issuer),**
  - **C-MAC (authentication of the issuer and integrity of the command),**
  - **C-DECRYPTION (authentication of the issuer, integrity and confidentiality of the command),**
  - **R-MAC (authentication of the card and integrity of the response),**
  - **R-ENCRYPTION (authentication of the card, integrity and confidentiality of the response).**
- **The Initial Chaining Vector (ICV) is a value used to compute the MAC value of a message, which relates it to the previous messages of the current session.**

The subjects have the following security attributes for SCP11 [GP23 Amend F]:

- **All 3 variants are supported, namely SCP11a, SCP11b, and SCP11c.**
- **An Elliptic Curve Key Agreement Algorithm (ECKA) is used for the establishment of session keys as described in BSI Technical Guideline TR-03111 [TR 03111].**
- **The Key Agreement Algorithm generates a shared secret that will be used in a key derivation process.**
- **It uses X9.63 Key Derivation Function for calculating cryptographic keys and session keys.**
- **SCP11a uses two pairs of static keys and two pairs of ephemeral keys as described in [GP23 Amend F].**
- **In SCP11b, it uses two ephemeral key pairs and one static key pair. The OCE does not have a static key pair. The ephemeral key pair of the OCE is used twice.**

- **In SCP11c: With two static key pairs and one ephemeral key pair. The SD does not create an ephemeral key pair. Instead, the static key pair of the SD is used twice. This scheme is described in [NIST 800-56A] as "One-Pass Unified Model, C(1e, 2s, ECC CDH) Scheme".**
- **There is a need for certificate verification through the CA or a Key Authority**
- **The Security Domain and OCE (Off-Card Entity) may need to traverse and verify a chain of certificates from established trust points down to each other's public key.**
- **The OCE has to retrieve the SD's certificate first.**
- **For SCP11a and SCP11c only: The OCE authenticates itself to the SD by providing a certificate signed by the CA-KLOC and by providing the first APDU after secure channel establishment with a correct MAC.**
- **The SD authenticates to the OCE by providing a certificate signed by the CA-KLCC and by generating a receipt at the end of the key establishment procedure.**
- **SCP11a and SCP11b provide Forward Secrecy provides forward secrecy while SCP11c does not.**
- **SCP11c can be used for session replay or scripting but some commands are forbidden (i.e. Put Key, Delete Key, Set Status, and Store Data that loads keys).**
- **For SCP11a and SCP11b, only two Security Levels for Secure Messaging are defined in this specification; the Security Level is set in the key usage qualifier data object of the MUTUAL or INTERNAL AUTHENTICATE command:**
  - **C-MAC and R-MAC only**
  - **C-DECRYPTION, R-ENCRYPTION, C-MAC, and R-MAC**
- **For SCP11c, four Security Levels for Secure Messaging are defined in this specification; the Security Level is set in the key usage qualifier data object of the MUTUAL AUTHENTICATE command:**
  - **C-MAC and R-MAC only**
  - **C-DECRYPTION, R-ENCRYPTION, C-MAC, and R-MAC**
  - **C-DECRYPTION and C-MAC**
  - **C-DECRYPTION, C-MAC, and R-MAC**
- **The MAC chaining value of the first APDU command after the MUTUAL or INTERNAL AUTHENTICATE command shall be set to the value of the receipt returned by the SD in the MUTUAL or INTERNAL AUTHENTICATE response.**

The subjects have the following security attributes for SCP21:

- **The nonce and challenge is a random number generated by the subject in order to identify the current session**
- **The token is a secret relative to the current smart card session that serves to authenticate the on- and off-card subjects. The token is derived from the known secret and/ or data exchanged of the on-card and off-card during authentication.**
- **The PACE password is used to derive key to encrypt the generated PACE nonce. The possible PACE passwords are: MRZ, CAN, PIN, PUK.**
- **The Chip Authentication static key pair is used in key agreement with off-card entity to derive shared secret as material for key derivation of Chip Authentication session key**
- **The Terminal Authentication CVCA public key is used to verify the DV and subsequently Terminal certificate. The public key extracted from verified Terminal certificate is used to verify the signature of Terminal and hence to authenticate the Terminal.**
- **The Session Keys is a set of keys derived from key agreement shared secret during PACE or Chip Authentication. It is used to verify the origin and integrity of the received message, and to encrypt/ decrypt their contents. This set is made of the following keys:**
  - **Session Secure Channel Encryption Key (K-ENC).**
  - **Session Secure Channel MAC Key (K-MAC).**
- **The Security Level defined for the messages that the card receives through the secure channel. The possible security levels are:**
  - **C-MAC C-DEC R-MAC R-ENC (authentication, integrity, and confidentiality of command and response).**
- **The Send Sequence Counter is a counter used to compute the MAC value of the message, which relates it to the previous messages of the current session. For AES, it is also used to compute the initialization vector for message encryption/ decryption**
- **The Current Privacy Status denotes a particular step reached in the execution of authentication. The possible privacy status are:**

- **PACE_MRZ**
- **PACE_CAN**
- **PACE_PIN**
- **PACE_PUK**
- **TA**
- **CA**
- **GAP**

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 [GP23]);**
  - **installation (Section 9.3.6 of [GP23]);**
  - **extradition (Section 9.4.1 of [GP23]);**
  - **registry update (Section 9.4.2 of [GP23]);**
  - **SD personalization rules, pull and push models [GP23 Amend A].**
- **Rule IFF-1: The SD may process a RECEIVE (INITIALIZE-UPDATE) operation only if the key set specified in the command exist in the SD and is integer.**
  Rule IFF-2: The ISD may process a RECEIVE (EXTERNAL-AUTHENTICATE) operation if the following conditions hold:
  - **The cryptogram received from the off-card subject is equal to the cryptogram computed by the Security Domain.**
  - **The MAC attached to the message has been generated from the CMAC session key and the current value of the ICV.**

  Rules IFF-3: The ISD may process a RECEIVE (GET-DATA) operation if the following condition holds: If the command security level is at least C-MAC, the MAC attached to the message has been generated from the command using the C-MAC session key and the current value of the ICV.
  Rules IFF-4: The ISD may process a RECEIVE (M) operation for any other command M different from the ones cited in the rules above if the following conditions hold:
  - **The current security level is at least AUTHENTICATED.**
  - **If the command security level is at least C-MAC, the MAC attached to the message has been generated from the clear-text command using the C-MAC session key and the current value of the ICV**.

FDP_IFF.1.3/SC The TSF shall enforce the **no additional information flow control SFP rules**.

FDP_IFF.1.4/SC The TSF shall explicitly authorize an information flow based on the following rules: **no additional information flow control SFP rules**.

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**.

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** to restrict the ability to **modify** the security attributes **(1) key set, Static keys, Command security Level, Secure channel protocol of a security domain, (2) Session Keys, Sequence Counter and ICV of a session (for SCP02, SCP03, SCP11 and SCP21), , (3) security domain parameters, connection parameters, security parameters, retry policy parameters,** to **(1 & 2 & 3 & 4) the actor associated with the security domain:**

- **The OEM for ISD,**
- **The Service Provider for SSD**,
- **The CA for CASD.**

Application note: the authorized identified roles could be the card issuer (off-card) or a SD (on-card).

FMT_MSA.3/SC	Static attribute initialization

FMT_MSA.3.1/SC The TSF shall enforce the **Secure Channel Protocol (SCP) information flow control policy** to provide **restrictive** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/SC The TSF shall allow the **authorized entities (OEM, or Application Provider)** 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.

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 [GP23]:

- **loading (Section 9.3.5 of [GP23]);**
- **installation (Section 9.3.6 of [GP23]);**
- **extradition (Section 9.4.1 of [GP23]);**
- **registry update (Section 9.4.2 of [GP23]);**
- **SD personalization rules, pull and push models [GP23 Amend A].**

The management functions are:

- **For SCP02 and SCP03**
  - **Generating a new card challenge during the setup of a Secure Channel.**
  - **Generating the session keys for the Secure Channel from the specified static key set and its associated Sequence Counter.**
  - **Generating the card cryptogram from the host and card challenges and the session keys.**
  - **Increasing by one the Sequence Counter associated to the specified Key Set upon successful opening a Secure Channel.**
  - **Setting the security level of the Secure Channel as the authenticated authorized entities (OEM, or Application Provider) had specified during its setup.**
  - **Updating the current value of the ICV upon reception of a new message through the Secure Channel.**
  - **On request of the Issuer or authorized application provider, loading or replacing the static keys that the associated Security Domain uses to open a Secure Channel.**

- **For SCP11**
  - **Retrieve the certificate of the other party: OCE from SD or SD from OCE**
  - **Verify certificates or certificate chains are signed by a Controlling Authority (CA), either CA-KLCC, or CA-KLOC**
  - **Generate an ephemeral key pair using Elliptic Curve Key Agreement (ECKA)**
  - **Calculates the shared secret from the generated key pair and the static key pairs according to ECKA**
  - **Derive AES session keys from the shared secret using X.963 key derivation function**
  - **Setting the security level of the Secure channel**
  - **Use SCP03 for secure messaging**
  - **Control the operations or commands within a session according to the security level and the variant of the SCP11 protocol**
  - **Manage Store Data command to store or replace a certificate, store or replace a certificate white list, or a CA-KLOC identifier**

- **For SCP21**
  - **Generating PACE random nonce**
  - **Generating Chip Authentication version 2 and Terminal Authentication random challenge**

- **Generating ephemeral keys for PACE key agreement**
- **Generating shared secret using key agreement**
- **Generating the session keys for the Secure Channel using key derivation function from the shared secret**
- **Generating the token to authenticate subject to external entity**
- **Verifying received token to authenticate external entity to subject**
- **Setting the security level of the Secure Channel**
- **Setting the Current Privacy Status**
- **Increasing by one the Sequence Counter every time before a command or response APDU is generated in Secure Messaging**
- **Verifying certificate chain rooted to subject stored CVCA and verifying signature during Terminal Authentication**
- **Verifying and storing of new CVCA**
- **Computing the effective authorization of the off-card entity during Terminal Authentication**

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;**
- **requesting data that identifies the card or the Card Issuer;**

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

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

Application note: the GlobalPlatform TSF mediated actions listed in [GP23] such as selecting an application, requesting data, initializing, etc.

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** 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.

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**.

7.2.2.3 OS Update, OS Configurability and Secure API Security Functional Requirements

This group includes TOE Security Requirements for OS update, Activation/deactivation of optional services and Secure API.

Application note: Activation of patch (OS Update) follows rules defined for optional platform service handled with GemActivate.

FMT_SMR.1/GemActivate Security roles

FMT_SMR.1.1/GemActivate The TSF shall maintain the roles **[GemActivate Administrator, OEM]**.

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

FMT_SMF.1/GemActivate Specification of Management Functions

FMT_SMF.1.1/GemActivate The TSF shall be capable of performing the following management functions: **activation of additional code**, **activation of optional platform service**
Application note: once verified and installed, additional code needs to be activated to become effective.

FMT_MOF.1/GemActivate Management of security functions behavior

FMT_MOF.1.1/GemActivate The TSF shall restrict the ability to **disable and enable** the functions **activation or inhibition of optional platform services as: cryptographic algorithm, package, applet instance, scalability (extension of available NVM) and NFC interface** to **GemActivate Administrator, OEM**.

FMT_MSA.1/GemActivate Management of security attributes

FMT_MSA.1.1/GemActivate The TSF shall enforce the **GemActivate access control SFP** to restrict the ability to **modify** the security attributes **state (deactivated, activated, inhibited) of optional platform service** to **GemActivate Administrator under control of OEM**.

FMT_MTD.1/GemActivate Management of TSF data

FMT_MTD.1.1/GemActivate The TSF shall restrict the ability to **query** the **[List of deactivated/activated/ inhibited optional platform services]** to **[GemActivate Administrator and OEM]**.

FIA_ATD.1/OS-UPDATE User attribute definition

FIA_ATD.1.1/OS-UPDATE The TSF shall maintain the following list of security attributes belonging to individual users:

- **Patch Package AID for each Patch package,**
- **Patch ID, Patch Activation Status for each patch.**

Refinement: "Individual users" stands for additional code.

FDP_ACC.1/GemActivate Subset access control

FDP_ACC.1.1/GemActivate The TSF shall enforce the **OS Update Access Control Policy** on:

- **Subjects: S.OS-Developer, S.INSTALLER, S.SD**
- **Objects: additional code and associated cryptographic signature**
- **Operations: loading, installation and activation of additional code**

Application note: S.OS-Developer is the TOE subject acting on behalf of the OS Developer.

FDP_ACF.1/GemActivate

FDP_ACF.1.1/GemActivate The TSF shall enforce the **OS Update Access Control Policy** to objects based on the following:

- **Subjects:**
  - **S.OS-Developer, responsible for verifying the signature of the additional code, and for decrypting it before authorizing its loading, installation and activation**
  - **S.INSTALLER, defined in [PP-JCS] and represented by the GlobalPlatform Environment (OPEN) on the card, the Card Life Cycle attributes (defined in Section 5.1.1 of [GP23]);**
  - **S.SD receiving the Card Content Management commands (through Load and Install APDUs)**

- **Objects: additional code and associated cryptographic signature**

- **Security Attributes:**
  - **The additional code cryptographic signature verification status**
  - **The Identification Data verification status (between the Initial TOE and the additional code)**

FDP_ACF.1.2/GemActivate The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

- **The verification of the additional code cryptographic signature (using D.OS-UPDATE_SGNVER-KEY) by S.OS-Developer is successful.**
- **The comparison between the identification data of both the Initial TOE and the additional code demonstrates that the OS Update operation can be performed.**

FDP_ACF.1.3/GemActivate The TSF shall explicitly authorize access of subjects to objects based on the following additional rules:
Rule GA-1: A loading and linking request for a package referencing restricted packages may be accepted only if the APDU command specifying the request contains a DAP well-formed according to [GP23] and its verification using GemActivate key by GemActivate Administrator is successful.

FDP_ACF.1.4/GemActivate The TSF shall explicitly deny access of subjects to objects based on the following additional rules:

- **Rule GA-2: When a loading and linking request for a package referencing restricted packages fails, package is not installed and associated NVM is recovered.**
- **Rule GA-3: When at least one of the rules for loading defined by GlobalPlatform [GP23] does not hold.**

Application notes:

- Identification data verification is necessary to ensure that the received additional code is actually targeting the TOE and that its version is compatible with the TOE version.
- Attributes of OS Update follow the same rules as optional platform services.

FMT_MSA.3/GemActivate Static attribute initialization

FMT_MSA.3.1/GemActivate The TSF shall enforce the **OS Update Access Control Policy** to provide **restrictive** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/GemActivate The TSF shall allow the **OS Developer** to specify alternative initial values to override the default values when an object or information is created.

Application note: the additional code signature verification status is set to “Fail” by default, therefore preventing any additional code from being installed until the additional code signature is actually successfully verified by the TOE.

FTP_TRP.1/OS-UPDATE

FTP_TRP.1.1/OS-UPDATE The TSF shall provide a communication path between itself and **remote** users that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from **disclosure**.

FTP_TRP.1.2/OS-UPDATE The TSF shall permit **remote users** to initiate communication via the trusted path.

FTP_TRP.1.3/OS-UPDATE The TSF shall require the use of the trusted path for **the transfer of the additional code to the TOE**.

The section below states the security functional requirements for the Secure API.

FPT_FLS.1/SecureAPI	Failure with preservation of secure state

FPT_FLS.1.1/SecureAPI The TSF shall preserve a secure state when the following types of failures occur: **the application fails to perform a specific execution flow control protected by the Secure API**.

FPT_ITT.1/SecureAPI	Basic internal TSF data transfer protection

FPT_ITT.1.1/SecureAPI The TSF shall protect TSF data from **disclosure and modification** when it is transmitted between separate parts of the TOE.

FPR_UNO.1/SecureAPI	Unobservability

FPR_UNO.1.1/SecureAPI The TSF shall ensure that **external attackers** are unable to observe the operation **as sensitive comparison or copy** on **sensitive objects defined** by **the application using the Secure API**.

7.2.2.4 *Global Privacy Framework*

This section on security functional requirements for the Global Privacy Framework is divided into sub-sections following the main security functionalities.

7.2.2.4.1 Security Functional Requirements for PACE and EAC2

Class FCS Cryptographic Support

The TOE shall meet the requirement "Cryptographic key generation (FCS_CKM.1)" as specified below (Common Criteria Part 2). The iterations are caused by different cryptographic key generation algorithms to be implemented and key to be generated by the TOE.

FCS_CKM.1/DH_PACE Cryptographic key generation – Diffie-Hellman for PACE and CA2 session keys

Hierarchical to:	No other components.
Dependencies:	[FCS_COP.1 Cryptographic operation]: fulfilled by **FCS_COP.1/PACE_ENC**, **FCS_COP.1/PACE_MAC** and **FCS_COP.1/PACE_CAM** FCS_CKM.4 Cryptographic key destruction: fulfilled **by** Error! Reference source not found.**/PACE**.
FCS_CKM.1.1 /DH_PACE	The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm [selection: *Diffie- Hellman-Protocol compliant to [PKCS3], ECDH compliant to [TR-03111]*] and specified cryptographic key sizes **Table 12~~Table 12~~ column Key size** bit that meet the following: [TR03110-2].

Key Usage	algorithm	Key size
/SKPICC, /PKPICC	**ECDH Key Agreement Algorithm –** [TR-03111] **DH Key Agreement Algorithm –** [PKCS3]	**ECDH: 192, 224, 256, 320, 384, 512, and 521 bits** **DH: 1024, 2048**
/TDESsession-ECDH /TDESsession-DH	**KDF –** [ICAO9303]	**112 bits**
/AESsession-ECDH /AESsession-DH	**KDF –** [ICAO9303]	**128, 192, 256 bits**

Table 12: FCS_CKM.1/DH_PACE iteration explanation

The TOE shall meet the requirement "Cryptographic key destruction (FCS_CKM.4)" as specified below (Common Criteria Part 2).

FCS_CKM.4/PACE Cryptographic key destruction

Hierarchical to:	No other components
Dependencies:	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or

FCS_CKM.1 Cryptographic key generation]: fulfilled by **FCS_CKM.1/DH_PACE**

FCS_CKM.4.1 /PACE	The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method [**Secure erasing of the value by overwriting the data with random numbers]** that meets the following: [**None]**.

FCS_COP.1/PACE_ENC Cryptographic operation – Encryption / Decryption AES

Hierarchical to:	No other components.
Dependencies:	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation]: fulfilled by **FCS_CKM.1/DH_PACE** FCS_CKM.4 Cryptographic key destruction: fulfilled by **FCS_CKM.4/PACE**.
FCS_COP.1.1 /PACE_ENC	The TSF shall perform secure messaging – encryption and decryption in accordance with a specified cryptographic algorithm **AES in CBC mode** and cryptographic key sizes **[Selection: 128, 192, 256 bits] Key size** that meet the following: [**TR03110-3**].

FCS_COP.1/PACE_MAC Cryptographic operation – CMAC

Hierarchical to:	No other components.
Dependencies:	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation]: fulfilled by **FCS_CKM.1/DH_PACE** FCS_CKM.4 Cryptographic key destruction: fulfilled by **FCS_CKM.4/PACE**
FCS_COP.1.1 /PACE_MAC	The TSF shall perform secure messaging – message authentication code in accordance with a specified cryptographic algorithm **Table 13~~Table 13~~ algorithm** and cryptographic key sizes **Table 13~~Table 13~~ Key size** that meet the following: compliant to [TR03110-3].

Algorithm explanation	algorithm	Key size	List of standards
/MAC_AES	**AES CMAC**	**128, 192, 256**	**TR03110-3**

Table 13: FCS_COP.1/PACE_MAC iteration explanation

FCS_COP.1/PACE_CAM Cryptographic operation – Modular Multiplication

Hierarchical to:	No other components.
Dependencies:	[FCS_CKM.1 Cryptographic key generation]: fulfilled by **FCS_CKM.1/DH_PACE** FCS_CKM.4 Cryptographic key destruction: fulfilled by **FCS_CKM.4/PACE**
FCS_COP.1.1 /PACE_CAM	The TSF shall perform modular multiplication with specify cryptography algorithm and cryptographic key sizes as in Table 14~~Table 14~~ Key size that meet the following: compliant to: [TR03110-1].

Algorithm type	algorithm	Key size
/CAM_ECDH	**ECC**	**192, 224, 256, 320, 384, 512, 521**

Table 14: FCS_COP.1/PACE_CAM iteration explanation

FCS_RNG.1/PACE Quality metric for random numbers

Hierarchical to: No other components

Dependencies: No dependencies

FCS_RNG.1.1 /PACE	The TSF shall provide a **hybrid deterministic** random number generator that implements: (DRG.4.1) The internal state of the RNG shall use PTRNG of class PTG.2 as random source. (DRG.4.2) The RNG provides forward secrecy. (DRG.4.3) The RNG provides backward secrecy even if the current internal state is known. (DRG.4.4) The RNG provides enhanced forward secrecy after calling the re-seed function that acts as a refreshing done at each random generation. (DRG.4.5) The internal state of the RNG is seeded by an internal entropy source, PTRNG of class PTG.2
FCS_RNG.1.2 /PACE	The TSF shall provide random numbers that meet: RGS [RGS-B1] and [AIS31] DRG3 & DRG4. (DRG.4.6) The RNG generates output for which 2^35 strings of bit length 128 are mutually different with probability equal to (1 – 1/2^58). (DRG.4.7) Statistical test suites cannot practically distinguish the random numbers from output sequences of an ideal RNG. The random numbers must pass test procedure A.

Application note: This SFR requires the TOE to generate random numbers used for the authentication protocols (i.e. PACE, CA2) as required by FIA_UAU.4.
Regarding the structure of this SFR, even if it is related to the PACE component, the structure comes from [PP-JCS-Open].

Class FIA Identification and Authentication

For the ease of presentation, we give an overview of the authentication mechanisms and directly corresponding SFRs.
The table below provides an overview on the authentication mechanisms used.

Name	SFR for the TOE
Chip Authentication Protocol v.1	**FIA_UAU.5/PACE**
Chip Authentication Protocol v.2	**FIA_API.1/CA, FIA_UAU.5/PACE, FIA_UAU.6/CA**
Terminal Authentication Protocol v.1	**FIA_UAU.5/PACE**
Terminal Authentication Protocol v.2	**FIA_UAU.1/EAC2_Terminal, FIA_UAU.5/PACE**
PACE protocol	

Table 15: Overview on authentication SFR

FIA_AFL.1/PACE Authentication failure handling – PACE authentication using non-blocking authorisation data

Hierarchical to: No other components.

Dependencies: FIA_UAU.1 Timing of authentication: fulfilled by FIA_UAU.1/PACE.

FIA_AFL.1.1 /PACE	The TSF shall detect when [**Number in Table 16~~Table 16~~**] unsuccessful authentication attempt occurs related to authentication attempts using the PACE password as shared password.
FIA_AFL.1.2 /PACE	When the defined number of unsuccessful authentication attempts has been met, the TSF shall [**Actions in Table 16~~Table 16~~**].

Password	Number	Actions
MRZ, CAN	**1**	**Exponentially increase time delay before new authentication attempt is possible**.
PIN	**6**	**Block PIN**.

Table 16: FIA_AFL.1/PACE refinements

FIA_UID.1/PACE Timing of identification

Hierarchical to: No other components
Dependencies: No dependencies

FIA_UID.1.1 /PACE — The TSF shall allow
1. to establish the communication channel,
2. carrying out the PACE Protocol according to [TR03110-2],
3. to read the Initialization Data if it is not disabled by TSF according to FMT_MTD.1/INI_DIS
4. to identify themselves by selection of the authentication key.

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

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

FIA_UAU.1/PACE Timing of authentication

Hierarchical to: No other components
Dependencies: FIA_UID.1 Timing of identification
fulfilled by **FIA_UID.1/PACE**

FIA_UAU.1.1 /PACE — The TSF shall allow
1. to establish the communication channel,
2. carrying out the PACE Protocol according to [TR03110-2],
3. to read the Initialization Data if it is not disabled by TSF according to FMT_MTD.1/INI_DIS
4. to identify themselves by selection of the authentication key.

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

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

FIA_UAU.4/PACE Single-use authentication mechanisms - Single-use authentication of the Terminal by the TOE

Hierarchical to: No other components
Dependencies: No dependencies

FIA_UAU.4.1 /PACE The TSF shall prevent reuse of authentication data related to
1. PACE Protocol according to [TR03110-2],
2. Authentication Mechanism based on **AES**
3. Terminal Authentication Protocol 2 according to [TR03110-2],
4. [**assignment: Terminal Authentication Protocol 1**]

Application note: For the PACE protocol, the TOE randomly selects nonce of 128 bits length being (almost) uniformly distributed.

FIA_UAU.5/PACE Multiple authentication mechanisms

Hierarchical to: No other components
Dependencies: No dependencies

FIA_UAU.5.1 /PACE The TSF shall provide
1. PACE Protocol according to [TR03110-2],
2. Passive Authentication according to [ICAO9303]
3. Secure messaging according to [TR03110-3],
4. Symmetric Authentication Mechanism based on **AES**
5. Terminal Authentication 2 protocol according to [TR03110-2]
6. Chip Authentication 2 according to [TR03110-2]
7. [assignment: Terminal Authentication 1, Chip Authentication 1]

To support user authentication.

FIA_UAU.5.2 /PACE The TSF shall authenticate any user's claimed identity according to the following rules:
1. TOE accepts the authentication attempt as Pre-personalization Agent by the Symmetric Authentication Mechanism with the Pre-personalization Agent Key.
2. Having successfully run the PACE protocol the TOE accepts only received commands with correct message authentication code sent by secure messaging with the terminal by the PACE protocol.
3. The TOE accepts the authentication attempt as Personalization Agent by the Symmetric Authentication Mechanism with Personalization Agent Key.
4. The TOE accepts the authentication attempt by means of the Terminal Authentication 2 protocol, only if (i) the terminal presents its static public key PK_{PCD} and the key is successfully verifiable up to the CVCA and (ii) the terminal uses the PICC identifier IDPICC = Comp(ephem-PKPICC-PACE) calculated during, and and the secure messaging established by the current PACE authentication.
5. Having successfully run Chip Authentication 2, the TOE accepts only received commands with correct message authentication codes sent by secure messaging with the key agreed with the terminal by Chip Authentication 2.

FIA_UAU.6/PACE Re-authenticating – Re-authenticating of Terminal by the TOE

Hierarchical to: No other components
Dependencies: No dependencies

FIA_UAU.6.1 /PACE The TSF shall re-authenticate the user under the conditions each command sent to the TOE after successful run of the PACE Protocol shall be verified as being sent by the PACE terminal.

Class FDP User Data Protection

The TOE shall meet the requirement “Subset residual information protection (FDP_RIP.1/PACE)” as specified below (Common Criteria Part 2).

FDP_RIP.1/PACE Subset residual information protection

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FDP_RIP.1.1 /PACE	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: 1. Session Keys (immediately after closing related communication session). 2. Ephemeral private key ephem - SK_{PICC}- PACE (by having generated a DH shared secret *K*). 3. Secret electronic document holder authentication data, e.g. PIN and/or PUK (when their temporarily stored values are not used any more)

Class FTP Trusted Path/Channels

FTP_ITC.1/PACE Inter-TSF trusted channel after PACE

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FTP_ITC.1.1 /PACE	The TSF shall provide a communication channel between itself and a PACE terminal 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. The trusted channel shall be established by performing the PACE protocol according to [TR03110-2].
FTP_ITC.1.2 /PACE	The TSF shall permit a PACE terminal to initiate communication via the trusted channel.
FTP_ITC.1.3 /PACE	The TSF shall enforce communication via the trusted channel for any data exchange between the TOE and a PACE terminal after PACE.

Class FMT Security Management

Application note: The SFR FMT_SMF.1 and FMT_SMR.1 provide basic requirements to the management of the TSF data.

The TOE shall meet the requirement “Specification of Management Functions (FMT_SMF.1)” as specified below (Common Criteria Part 2).

FMT_SMF.1/PACE Specification of Management Functions

Hierarchical to:	No other components
Dependencies:	No dependencies
FMT_SMF.1.1 /PACE	The TSF shall be capable of performing the following management functions: 1. Initialization. 2. Pre-personalization 3. Personalization 4. Configuration

The TOE shall meet the requirement “Security roles (FMT_SMR.1)” as specified below (Common Criteria Part 2).

FMT_SMR.1/PACE Security roles

Hierarchical to:	No other components
Dependencies:	FIA_UID.1 Timing of identification fulfilled by **FIA_UID.1/PACE, FIA_UID.1/EAC2_Terminal**
FMT_SMR.1.1 /PACE	The TSF shall maintain the roles 1. Terminal, 2. PACE Terminal 3. EAC2 Terminal [Authentication terminal] 4. Manufacturer 5. Personalization Agent
FMT_SMR.1.2 /PACE	The TSF shall be able to associate users with roles.

The TOE shall meet the requirement “Limited capabilities (FMT_LIM.1)” as specified below (Common Criteria Part 2 extended).

FMT_LIM.1/PERSO Limited capabilities

Hierarchical to:	No other components
Dependencies:	FMT_LIM.2 Limited capabilities: fulfilled by **FMT_LIM.2/PERSO**
FMT_LIM.1.1 /PERSO	The TSF shall be designed in a manner that limits their capabilities so that in conjunction with “Limited availability (FMT_LIM.2)” the following policy is enforced: Deploying test features after TOE delivery do not allow 1.User Data to be manipulated and disclosed, 2.TSF data to be manipulated or disclosed, 3.software to be reconstructed, 4. Substantial information about construction of TSF to be gathered which may enable other attacks.

The TOE shall meet the requirement “Limited availability (FMT_LIM.2)” as specified below (Common Criteria Part 2 extended).

FMT_LIM.2/PERSO Limited availability

Hierarchical to:	No other components
Dependencies:	FMT_LIM.1 Limited capabilities: fulfilled by **FMT_LIM.1/PERSO**
FMT_LIM.2.1 /PERSO	The TSF shall be designed in a manner that limits their availability so that in conjunction with “Limited capabilities (FMT_LIM.1)” the following policy is enforced: Deploying Test Features after TOE Delivery does not allow 1.User Data to be manipulated and disclosed, 2.TSF data to be manipulated or disclosed, 3.software to be reconstructed, 4.substantial information about construction of TSF to be gathered which may enable other attacks

Application note: The term “software” of FMT_LIM.1.1 and FMT_LIM.2.1 refers to both IC Dedicated and IC Embedded Software.

The TOE shall meet the requirement “Management of TSF data (FMT_MTD.1)” as specified below (Common Criteria Part 2). The iterations address different management functions and different TSF data.

FMT_MTD.1/INI_ENA Management of TSF data – Writing of Initialization Data and Pre-personalization Data

Hierarchical to:	No other components
Dependencies:	FMT_SMF.1 Specification of management functions: fulfilled by **FMT_SMF.1/PACE** FMT_SMR.1 Security roles: fulfilled by **FMT_SMR.1/PACE.**
FMT_MTD.1.1/ INI_ENA	The TSF shall restrict the ability to write the Initialization Data and Pre-personalization Data to the Manufacturer.

Application note: The pre-personalization Data includes but is not limited to the authentication reference data for the Personalization Agent which is the symmetric cryptographic Personalization Agent Key.

FMT_MTD.1/INI_DIS Management of TSF data – Disabling of Read Access to Initialization Data and Pre-personalization Data

Hierarchical to:	No other components
Dependencies:	FMT_SMF.1 Specification of management functions: fulfilled by **FMT_SMF.1/PACE** FMT_SMR.1 Security roles: fulfilled by **FMT_SMR.1/PACE.**
FMT_MTD.1.1/ INI_DIS	The TSF shall restrict the ability to read out the Initialisation Data and the Pre-personalisation Data to the Personalisation Agent

FMT_MTD.1/KEY_READ Management of TSF data – Private Key Read

Hierarchical to:	No other components
Dependencies:	FMT_SMF.1 Specification of management functions: fulfilled by **FMT_SMF.1/PACE** FMT_SMR.1 Security roles: fulfilled by **FMT_SMR.1/PACE**.
FMT_MTD.1.1/ KEY_READ	The TSF shall restrict the ability to read the 1. PACE passwords, 2. Personalization Agent Keys, 3. the Chip Authentication private key(s) (SK_{PICC}) 4. the Restricted Identification private key(s) to none.

Class FPT Protection of the Security Functions

The TOE shall prevent inherent and forced illicit information leakage for User Data and TSF Data. The security functional requirement FPT_EMS.1 addresses the inherent leakage. With respect to the forced leakage they

have to be considered in combination with the security functional requirements “Failure with preservation of secure state (FPT_FLS.1)” and “TSF testing (FPT_TST.1)” on the one hand and “Resistance to physical attack (FPT_PHP.3)” on the other. The SFRs “Limited capabilities (FMT_LIM.1)”, “Limited availability (FMT_LIM.2)” and “Resistance to physical attack (FPT_PHP.3)” together with the SAR “Security architecture description” (ADV_ARC.1) prevent bypassing, deactivation and manipulation of the security features or misuse of TOE security functionality.

The TOE shall meet the requirement “TOE Emanation (FPT_EMS.1)” as specified below (Common Criteria Part 2 extended):

FPT_EMS.1 TOE Emanation

Hierarchical to: No other components
Dependencies: No dependencies.

FPT_EMS.1.1 The TOE shall not emit **electromagnetic and current emissions** in excess of **intelligible threshold** enabling access to

1. the session keys (PACE-K MAC, PACE-KEnc), (CA-K MAC, CA-KEnc)
2. the ephemeral private key ephem - SK PICC- PACE,
3. the Chip Authentication private keys (SK-PICC)
4. the PIN, PUK
5. Personalization Agent Key(s)
6. Applicative keys and sensitive data

FPT_EMS.1.2 The TSF shall ensure any users are unable to use the following interface TOE external interfaces available according to form factor to gain access to

1. the session keys (PACE-K MAC, PACE-KEnc), (CA-K MAC, CA-KEnc
2. the ephemeral private key ephem - SK PICC- PACE
3. The Chip uthenticate private keys (SK-PICC)
4. The PIN, PUK
5. Personalization Agent Key(s)
6. Applicative keys and sensitive data

Application note: When application is MTRD, Applicative keys are **Chip Authentication Private Key** and **Active Authentication Key**, and sensitive data are **EF.DG3 and EF.DG4**.

The following security functional requirements address the protection against forced illicit information leakage including physical manipulation.

FPT_FLS.1 Failure with preservation of secure state

Hierarchical to: No other components
Dependencies: No dependencies.

FPT_FLS.1.1 The TSF shall preserve a secure state when the following types of failures occur:

1. Exposure to operating conditions causing a TOE malfunction,
2. Failure detected by TSF according to FPT_TST.1.

The TOE shall meet the requirement “TSF testing (FPT_TST.1)” as specified below (Common Criteria Part 2).

FPT_TST.1 TSF testing

Hierarchical to: No other components
Dependencies: No dependencies.

FPT_TST.1.1 The TSF shall run a suite of self-tests **[Description of self-tests** in table below**]** to demonstrate the correct operation of the TSF.

FPT_TST.1.2 The TSF shall provide authorised users with the capability to verify the integrity of TSF data.

FPT_TST.1.3 The TSF shall provide authorised users with the capability to verify the integrity of stored TSF executable code.

Conditions under which self-test should occur	Description of self-tests
During initial start-up	**RNG live test, sensor test, FA detection, Integrity Check of NVM ES**
Periodically	**RNG monitoring, FA detection**
After cryptographic computation	**FA detection**
Before any use or update of TSF data	**FA detection, Integrity Check of related TSF data**

Table 17: FPT_TST triggering conditions

The TOE shall meet the requirement “Resistance to physical attack (FPT_PHP.3)” as specified below (Common Criteria Part 2).

FPT_PHP.3 Resistance to physical attack

Hierarchical to: No other components
Dependencies: No dependencies.

FPT_PHP.3.1 The TSF shall resist physical manipulation and physical probing to the TSF by responding automatically such that the SFRs are always enforced.

7.2.2.4.2 Security Functional Requirements for EAC2

Class FCS Cryptographic Support

FCS_COP.1/SHA Cryptographic operation – Hash for key derivation

Hierarchical to: No other components.

Dependencies:
[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation] not fulfilled, but **justified:** A hash function does not use any cryptographic key; hence, neither a respective key import nor key generation can be expected here. FCS_CKM.4 Cryptographic key destruction not fulfilled, but **justified:** A hash function does not use any cryptographic key; hence, a respective key destruction cannot be expected here.

FCS_COP.1.1/SHA
The TSF shall perform hashing in accordance with a specified cryptographic algorithm [**SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512**] and cryptographic key sizes [**none**] that meet the following: [**FIPS180-4**].
Application Note: For compressing (hashing) an ephemeral public key for DH (TA2 and CA2), the hash function SHA-1 shall be used ([TR03110-3]). The TOE shall implement as hash functions either SHA-1 or SHA-224 or SHA-256 for Terminal Authentication 2, cf. [TR03110-3].Within the normative Appendix of [TR03110-3] ‘Key Derivation Function’, it is stated that the hash function SHA-1 shall be used for deriving 128-bit AES keys, whereas SHA-256 shall be used for deriving 192-bit and 256-bit AES keys.

FCS_COP.1/SIG_VER Cryptographic operation – Signature verification

Hierarchical to:
No other components.

Dependencies:
[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation] not fulfilled, but **justified:** The root

key PK_{CVCA} (initialization data) used for verifying the DV Certificate is stored in the TOE during its personalization in the card issuing life cycle phase7. Since importing the respective certificates (Terminal Certificate, DV Certificate) does not require any special security measures except those required by the current SFR (cf. FMT_MTD.3 below), the current PP does not contain any dedicated requirement like FDP_ITC.2 for the import function.FCS_CKM.4 Cryptographic key destruction not fulfilled, but **justified:** Cryptographic keys used for the purpose of the current SFR (PK_{PCD}, PK_{DV}, PK_{CVCA}) are public keys; they do not represent any secret, and hence need not to be destroyed.

FCS_COP.1.1/SIG_VER

The TSF shall perform digital signature verification8 in accordance with a specified cryptographic algorithm [See table below] and cryptographic key sizes [See table below] that meet the following: [See table below].

Application Note: This SFR is concerned with Terminal Authentication 2, cf. [TR03110-2].

Algorithm	Key size	List of standards
id-TA-RSA-PSS-SHA-1	Up to 3072	**PKCS#1**
id-TA-RSA-PSS-SHA-256	Up to 3072	**PKCS#1**
id-TA-RSA-PSS-SHA-512	Up to 3072	**PKCS#1**
id-TA-ECDSA-SHA-1	Up to EC-521	**TR-03111**
id-TA-ECDSA-SHA-224	Up to EC-521	**TR-03111**
id-TA-ECDSA-SHA-256	Up to EC-521	**TR-03111**
id-TA-ECDSA-SHA-384	Up to EC-521	**TR-03111**
id-TA-ECDSA-SHA-512	Up to EC-521	**TR-03111**

Table 18: FCS_COP.1.1/SIG_VER iteration explanation

Class FIA Identification and Authentication

FIA_API.1/CA Authentication Proof of Identity

Hierarchical to:

No other components.

Dependencies:

No dependencies.

FIA_API.1.1/CA

The TSF shall provide the protocol Chip Authentication 2 according to [TR03110-2], to prove the identity of the TOE.

FIA_UID.1/EAC2_Terminal Timing of identification

Hierarchical to:

No other components.

Dependencies:

No dependencies.

FIA_UID.1.1/EAC2_Terminal

The TSF shall allow

1. To establish a communication channel

2. Carrying out the PACE protocol according to [TR03110-2]

3. To read the Initialization Data if it is not disabled by TSF according to FMT_MTD.1/INI_DIS

4. Carrying out the Terminal Authentication protocol 2 according to [TR03110-2]

5. [None]

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

FIA_UID.1.2/EAC2_Terminal

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

Application Note: The user identified after a successfully performed TA2 is an EAC2 terminal. The types of EAC2 terminals are application dependent;
Application Note: In the life cycle phase manufacturing, the manufacturer is the only user role known to the TOE. The manufacturer writes the initialization data and/or pre-personalization data in the audit records of the IC. Note that a personalization agent acts on behalf of the electronic document issuer under his and the CSCA's and DS's policies. Hence, they define authentication procedures for personalization agents. The TOE must functionally support these authentication procedures. These procedures are subject to evaluation within the assurance components ALC_DEL.1 and AGD_PRE.1. The TOE assumes the user role personalization agent, if a terminal proves the respective Terminal Authorization level (e. g. a privileged terminal, cf. [TR03110-2]).

FIA_UAU.1/EAC2_Terminal Timing of authentication

Hierarchical to:

No other components.

Dependencies:

FIA_UID.1 Timing of identification fulfilled by FIA_UID.1/EAC2_Terminal

FIA_UAU.1.1/EAC2_Terminal

The TSF shall allow

1. To establish a communication channel,

2. Carrying out the PACE protocol according to [TR03110-2],

3. To read the Initialization Data if it is not disabled by TSF according to FMT_MTD.1/INI_DIS

4. Carrying out the Terminal Authentication 2 protocol according to [TR03110-2]

On behalf of the user to be performed before the user is authenticated.

FIA_UAU.1.2/EAC2_Terminal

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

Application Note: The user authenticated after a successful run of TA2 is an EAC2 terminal. The authenticated terminal will immediately perform Chip Authentication 2 as required by FIA_API.1/CA using, amongst other, Comp(ephem-PK$_{PCD}$-TA) from the accomplished TA2.

FIA_UAU.6/CA Re-authenticating of Terminal by the TOE

Hierarchical to:
No other components.

Dependencies:
No dependencies.

FIA_UAU.6.1/CA
The TSF shall re-authenticate the user under the conditions ~~-~~ each command sent to the TOE after a successful run of Chip Authentication 2 shall be verified as being sent by the EAC2 terminal.

Class FTP Trusted Path/Channels

FTP_ITC.1/CA2 Inter-TSF trusted channel after CA2

Hierarchical to:
No other components.

Dependencies:
No dependencies.

FTP_ITC.1.1/CA2
The TSF shall provide a communication channel between itself and **an EAC2 terminal** 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. **The trusted channel shall be established by performing the CA2 protocol according to [TR03110-2].**

FTP_ITC.1.2/CA2

The TSF shall permit **an EAC2 terminal** to initiate communication via the trusted channel.

FTP_ITC.1.3/CA2
The TSF shall enforce communication via the trusted channel for any data exchange between the TOE and an EAC2 terminal after Chip Authentication 2.

Application Note 32: The trusted channel is established after successful performing the PACE protocol (FIA_UAU.1/PACE), the TA2 protocol (FIA_UAU.1/EAC2_Terminal) and the CA2 protocol (FIA_API.1/CA). If Chip Authentication 2 was successfully performed, secure messaging is immediately restarted using the derived session keys (CA-K$_{MAC}$, CA-K$_{Enc}$). This secure messaging enforces the required properties of operational trusted channel; the cryptographic primitives being used for the secure messaging are as required by FCS_COP.1/PACE_ENC and FCS_COP.1/PACE_MAC.

Class FMT Security Management

FMT_MTD.1/Initialize_PIN Management of TSF data – Initialize PIN

Hierarchical to:

No other components.

Dependencies:
FMT_SMF.1 Specification of management functions fulfilled by **FMT_SMF.1**
FMT_SMR.1 Security roles fulfilled by **FMT_SMR.1/PACE**

FMT_MTD.1.1/Initialize_PIN
The TSF shall restrict the ability to write the initial PIN and PUK to the personalization agent.

7.3 SECURITY ASSURANCE REQUIREMENTS

The security assurance requirement level is EAL4 augmented with AVA_VAN.5 and ALC_DVS.2.

7.4 SECURITY REQUIREMENTS RATIONALE

7.4.1 TOE security objectives coverage for JCS, GP, OP Update and OS Configurability– Mapping table

7.4.1.1 TOE security objectives coverage – Mapping table

	O.CARD-MANAGEMENT	O.DOMAIN-RIGHTS	O.APPLI-AUTH	O.COMM_AUTH	O.COMM_INTEGRITY	O.COMM_CONFIDENTIALITY	O.SCP-SUPPORT	O.SID	O.FIREWALL	O.GLOBAL_ARRAYS_CONFID	O.GLOBAL_ARRAYS_INTEG	O.NATIVE	O.OPERATE	O.REALLOCATION	O.RESOURCES	O.ALARM	O.CIPHER	O.KEY-MNGT	O.PIN-MNGT	O.TRANSACTION	O.OBJ-DELETION	O.DELETION	O.LOAD	O.INSTALL	O.SCP.IC	O.SCP.RECOVERY	O.Secure_API	O.RNG	O.JCAPI-Services	O.REMOTE_SERVICE_AUDIT	O.REMOTE_SERVICE_ACTIVATION	O.Secure_Load_ACode	O.Secure_AC_Activation	O.TOE_Identification	O.CONFID-OS-UPDATE.LOAD
FDP_UIT.1/CCM	X																														X	X		
FDP_ROL.1/CCM	X		X																												X	X	X	
FDP_ITC.2/CCM	X																														X	X	X	
FPT_FLS.1/CCM	X		X																												X	X		
FCS_COP.1/DAP			X																												X			
FDP_ACC.1/SD	X	X																													X			
FDP_ACF.1/SD	X	X																													X			
FMT_MSA.1/SD	X	X																													X			
FMT_MSA.3/SD	X	X																													X			
FMT_SMF.1/SD	X	X																													X			
FMT_SMR.1/SD	X	X		X	X	X																									X			
FTP_ITC.1/SC	X	X		X	X	X																									X			
FCO_NRO.2/SC	X	X																													X			
FDP_IFC.2/SC	X	X		X	X	X																									X			
FDP_IFF.1/SC	X	X		X	X	X																									X			
FMT_MSA.1/SC	X	X		X	X	X																									X			

FMT_MSA.3/SC	X	X		X	X	X																								X			
FMT_SMF.1/SC	X	X			X	X																								X			
FIA_UID.1/SC	X	X		X																										X			
FIA_UAU.1/SC	X	X		X																										X			
FIA_UAU.4/SC	X	X																												X			
FDP_ACC.2/FIREWALL								X				X					X																
FDP_ACF.1/FIREWALL								X			X	X					X																
FDP_IFC.1/JCVM								X	X	X																							
FDP_IFF.1/JCVM								X	X	X																							
FDP_RIP.1/OBJECTS													X				X	X	X														
FMT_MSA.1/JCRE							X	X																									
FMT_MSA.1/JCVM							X	X																									
FMT_MSA.2/FIREWALL_JCVM								X																									
FMT_MSA.3/FIREWALL							X	X																									
FMT_MSA.3/JCVM							X	X																									
FMT_SMF.1								X						X																			
FMT_SMR.1								X						X																			
FCS_CKM.1/TDES																X	X																
FCS_CKM.1/AES																X	X																
FCS_CKM.1/RSA																X	X																
FCS_CKM.1/ECDSA																X	X																
FCS_CKM.1/HMAC																X	X																
FCS_CKM.1/ECPF																X	X																
FCS_CKM.1/ECDH																X	X																
FCS_CKM.1/DHGen																X	X																
FCS_CKM.4																X	X																
FCS_COP.1/TDES_CIPHER																X	X																
FCS_COP.1/TDES_MAC																	X																
FCS_COP.1/AES_CIPHER																X	X																
FCS_COP.1/AES_MAC																	X																
FCS_COP.1/RSA_SIGN																	X																
FCS_COP.1/RSA_CIPHER																X	X																
FCS_COP.1/ECDSA_SIGN																	X																
FCS_COP.1/ECDH																	X																
FCS_COP.1/Hash																															X			
FCS_COP.1/HMAC																	X																
FCS_COP.1/ECDSA_KEY_GEN																	X																
FCS_COP.1/DH_KEY_GEN																	X																
FCS_RNG.1																														X				
FDP_RIP.1/ABORT													X				X	X	X														
FDP_RIP.1/APDU									X				X				X	X	X														
FDP_RIP.1/GlobalArray									X				X				X	X	X														
FDP_RIP.1/bArray									X				X				X	X	X														
FDP_RIP.1/KEYS													X				X	X	X														

FDP_RIP.1/TRANSIENT													X				X	X	X																
FDP_ROL.1/FIREWALL												X		X				X	X																
FAU_ARP.1												X		X	X																				
FDP_SDI.2/DATA																	X	X																	
FPR_UNO.1																X	X	X																	
FPT_FLS.1/JCS												X		X	X										X										
FPT_TDC.1												X																							
FIA_ATD.1/AID								X				X																							
FIA_UID.2/AID								X																											
FIA_USB.1/AID								X				X																							
FMT_MTD.1/JCRE								X	X					X																					
FMT_MTD.3/JCRE								X	X					X																					
FDP_ITC.2/Installer								X	X			X														X									
FMT_SMR.1/Installer									X					X																					
FPT_FLS.1/Installer												X		X	X										X										
FPT_RCV.3/Installer												X		X								X		X											
FDP_ACC.2/ADEL																						X													
FDP_ACF.1/ADEL																						X													
FDP_RIP.1/ADEL													X				X	X	X			X													
FMT_MSA.1/ADEL								X	X													X													
FMT_MSA.3/ADEL								X	X													X													
FMT_SMF.1/ADEL								X	X					X																					
FMT_SMR.1/ADEL									X					X								X													
FPT_FLS.1/ADEL												X		X	X							X													
FDP_RIP.1/ODEL													X				X	X	X	X															
FPT_FLS.1/ODEL												X		X	X					X															
FCO_NRO.2/CM																						X													
FDP_IFC.2/CM																						X													
FDP_IFF.1/CM																						X													
FDP_UIT.1/CM																						X													
FIA_UID.1/CM																						X													
FMT_MSA.1/CM								X	X																										
FMT_MSA.3/CM								X	X																										
FMT_SMF.1/CM								X	X					X																					
FMT_SMR.1/CM									X					X																					
FTP_ITC.1/CM																						X													
FPT_TST.1/SCP							X																												
FPT_RCV.3/SCP																										X									
FPT_PHP.3/SCP																									X										
FPT_RCV.4/SCP							X																												
FPT_FLS.1/SecureAPI																											X								
FPT_ITT.1/SecureAPI																											X								
FPR_UNO.1/SecureAPI																											X								
FMT_SMR.1/GemActivate																														X	X	X	X		

FMT_SMF.1/GemActivate																													**X**	**X**	**X**	**X**		
FMT_MOF.1/GemActivate																														**X**	**X**	**X**		
FMT_MSA.1/GemActivate																														**X**	**X**	**X**		
FMT_MTD.1/GemActivate																													**X**					
FDP_ACC.1/GemActivate																															**X**			
FDP_ACF.1/ GemActivate																															**X**			
FMT_MSA.3/GemActivate																															**X**			
FTP_TRP.1/OS-UPDATE																																		**X**
FIA_ATD.1/OS-UPDATE																																	**X**	

Table 19: TOE Security Objectives coverage by Security Functional Requirements – Mapping table

7.4.1.2 TOE security objectives coverage – Rationale

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, and 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.

O.DOMAIN-RIGHTS
The security objective O.DOMAIN-RIGHTS is met by the following SFRs:

- 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, and 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.

O.APPLI-AUTH
The security objective O.APPLI-AUTH is met by the following SFRs:

- FDP_ROL.1/CCM ensures that card management operations may be cleanly aborted.
- FPT_FLS.1/CCM preserves a secure state when failures occur.
- FCS_COP.1/DAP ensures that the loaded Executable Application is legitimate by specifying the algorithm to be used in order to verify the DAP signature of the Verification Authority.

O.COMM_AUTH
This security objective is covered by the following security functional requirements:

- FTP_ITC.1/SC which ensures the origin 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 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.
- 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 platform receives.
- 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.

O.COMM_INTEGRITY
This security objective is covered by the following security functional requirements:

- FTP_ITC.1/SC which ensures the integrity of card management commands.

- FMT_SMF.1/SC specifies the actions activating the integrity check on the card management commands.
- FMT_SMR.1/SD defines the roles enabling to send and authenticate the card management requests for which the integrity has to be ensured.
- FDP_IFC.2/SC and FDP_IFF.1/SC enforces the Secure Channel Protocol information flow control policy to guarantee the integrity of administration requests.
- FMT_MSA.1/SC and FMT_MSA.3/SC covers indirectly this security objective by specifying security attributes enabling to guarantee the integrity of card management requests.
- 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 ensure the integrity of the card management commands.

O.COMM_CONFIDENTIALITY
This security objective is covered by the following security functional requirements:
- FTP_ITC.1/SC which ensures the confidentiality of card management commands.
- FMT_SMF.1/SC specifies the actions ensuring the confidentiality of the card management commands.
- FMT_SMR.1/SD defines the roles enabling to send and authenticate the card management requests for which the confidentiality has to be ensured.
- FDP_IFC.2/SC and FDP_IFF.1/SC enforces the Secure Channel Protocol information flow control policy to guarantee the confidentiality of administration requests.
- FMT_MSA.1/SC and FMT_MSA.3/SC covers indirectly this security objective by specifying security attributes enabling to guarantee the confidentiality of card management requests by decrypting those requests and imposing management conditions on that attributes.
- 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 ensure the confidentiality of the card management commands.

O.SCP-SUPPORT
The SCP is a part of the TOE supporting TSFs of the upper layer of the TOE, especially for recovery operations as FPT_RCV.4/SCP. This security objective is also covered by FPT_TST.1/SCP.

O.SID
Subjects' identity is AID-based (applets, packages), and is met by the following SFRs: FDP_ITC.2/Installer, FIA_ATD.1/AID, FMT_MSA.1/JCRE, FMT_MSA.1/JCVM, FMT_MSA.1/ADEL, FMT_MSA.1/CM, FMT_MSA.3/ADEL, FMT_MSA.3/FIREWALL, FMT_MSA.3/JCVM, FMT_MSA.3/CM, FMT_SMF.1/CM, FMT_SMF.1/ADEL, FMT_MTD.1/JCRE and FMT_MTD.3/JCRE. Lastly, installation procedures ensure protection against forgery (the AID of an applet is under the control of the TSFs) or re-use of identities (FIA_UID.2/AID, FIA_USB.1/AID).

O.FIREWALL
This objective is met by the FIREWALL access control policy FDP_ACC.2/FIREWALL and FDP_ACF.1/FIREWALL, the JCVM information flow control policy (FDP_IFF.1/JCVM, FDP_IFC.1/JCVM) and the functional requirement FDP_ITC.2/Installer. The functional requirements of the class FMT (FMT_MTD.1/JCRE, FMT_MTD.3/JCRE, FMT_SMR.1/Installer, FMT_SMR.1, FMT_SMF.1, FMT_SMR.1/ADEL, FMT_SMF.1/ADEL, FMT_SMF.1/CM, FMT_MSA.1/CM, FMT_MSA.3/CM, FMT_SMR.1/CM, FMT_MSA.2/FIREWALL_JCVM, FMT_MSA.3/FIREWALL, FMT_MSA.3/JCVM, FMT_MSA.1/ADEL, FMT_MSA.3/ADEL, FMT_MSA.1/JCRE, FMT_MSA.1/JCVM) also indirectly contribute to meet this objective.

O.GLOBAL_ARRAYS_CONFID
Only arrays can be designated as global, and the only global arrays required in the Java Card API are the APDU buffer, the global byte array input parameter (bArray) to an applet's install method and the global arrays created by the JCSystem.makeGlobalArray(…) method. The clearing requirement of these arrays is met by (FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray and FDP_RIP.1/bArray respectively). The JCVM information flow control policy (FDP_IFF.1/JCVM, FDP_IFC.1/JCVM) prevents an application from keeping a pointer to a shared buffer, which could be used to read its contents when the buffer is being used by another application.

O.GLOBAL_ARRAYS_INTEG

This objective is met by the JCVM information flow control policy (FDP_IFF.1/JCVM, FDP_IFC.1/JCVM), which prevents an application from keeping a pointer to the APDU buffer of the card, to the global byte array of the applet's install method or to the global arrays created by the JCSystem.makeGlobalArray(…) method. Such a pointer could be used to access and modify it when the buffer is being used by another application.

O.NATIVE
This security objective is covered by FDP_ACF.1/FIREWALL: the only means to execute native code is the invocation of a Java Card API method. This objective mainly relies on the environmental objective OE.APPLET, which uphold the assumption A.APPLET.

O.OPERATE
The TOE is protected in various ways against applets' actions (FPT_TDC.1), the FIREWALL access control policy FDP_ACC.2/FIREWALL and FDP_ACF.1/FIREWALL, and is able to detect and block various failures or security violations during usual working (FPT_FLS.1/ADEL, FPT_FLS.1/JCS, FPT_FLS.1/ODEL, FPT_FLS.1/Installer, FAU_ARP.1). Its security-critical parts and procedures are also protected: safe recovery from failure is ensured (FPT_RCV.3/Installer), applets' installation may be cleanly aborted (FDP_ROL.1/FIREWALL), communication with external users and their internal subjects is well-controlled (FDP_ITC.2/Installer, FIA_ATD.1/AID, FIA_USB.1/AID) to prevent alteration of TSF data (also protected by components of the FPT class). Almost every objective and/or functional requirement indirectly contributes to this one too.

Application note: Startup of the TOE (TSF-testing) can be covered by FPT_TST.1. This SFR component is not mandatory in [JCRE3], but appears in most of security requirements documents for masked applications. Testing could also occur randomly. Self-tests may become mandatory in order to comply with FIPS certification [FIPS 140-2].

O.REALLOCATION
This security objective is satisfied by the following SFRs: FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray, FDP_RIP.1/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/TRANSIENT, FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, and FDP_RIP.1/ADEL, which imposes that the contents of the re-allocated block shall always be cleared before delivering the block.

O.RESOURCES
The TSFs detects stack/memory overflows during execution of applications (FAU_ARP.1, FPT_FLS.1/ADEL, FPT_FLS.1, FPT_FLS.1/ODEL, and FPT_FLS.1/Installer). Failed installations are not to create memory leaks (FDP_ROL.1/FIREWALL, FPT_RCV.3/Installer) as well. Memory management is controlled by the TSF (FMT_MTD.1/JCRE, FMT_MTD.3/JCRE, FMT_SMR.1/Installer, FMT_SMR.1, FMT_SMF.1 FMT_SMR.1/ADEL, FMT_SMF.1/ADEL, FMT_SMF.1/CM and FMT_SMR.1/CM).

O.ALARM
This security objective is met by FPT_FLS.1/Installer, FPT_FLS.1, FPT_FLS.1/ADEL, FPT_FLS.1/ODEL which guarantee that a secure state is preserved by the TSF when failures occur, and FAU_ARP.1 which defines TSF reaction upon detection of a potential security violation.

O.CIPHER
This security objective is directly covered by FCS_CKM.1/TDES, FCS_CKM.1/AES, FCS_CKM.1/RSA, FCS_CKM.1/ECDSA, FCS_CKM.1/HMAC, FCS_CKM.1/ECPF, FCS_CKM.1/ECDH, FCS_CKM.1/DHGen, FCS_CKM.4, FCS_COP.1/TDES_CIPHER, FCS_COP.1/AES_CIPHER and FCS_COP.1/RSA_CIPHER. The SFR FPR_UNO.1 contributes in covering this security objective and controls the observation of the cryptographic operations which may be used to disclose the keys.

O.KEY-MNGT
This relies on the same security functional requirements as O.CIPHER, plus FDP_RIP.1 and FDP_SDI.2 as well. Precisely it is met by the following components: FCS_CKM.1/TDES, FCS_CKM.1/AES, FCS_CKM.1/RSA, FCS_CKM.1/ECDSA, FCS_CKM.1/HMAC, FCS_CKM.1/ECPF, FCS_CKM.1/ECDH, FCS_CKM.1/DHGen, FCS_CKM.4, FCS_COP.1/TDES_MAC, FCS_COP.1/AES_MAC, FCS_COP.1/RSA_SIGN, FCS_COP.1/ECDSA_SIGN, FCS_COP.1/ECDH, FCS_COP.1/RSA_CIPHER, FCS_COP.1/HMAC, FCS_COP.1/TDES_CIPHER, FCS_COP.1/ECDSA_KEY_GEN, FCS_COP.1/DH_KEY_GEN, FPR_UNO.1, FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/APDU,

FDP_RIP.1/GlobalArray, FDP_RIP.1/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL and FDP_RIP.1/TRANSIENT.

O.PIN-MNGT
This security objective is ensured by FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray, FDP_RIP.1/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL, FDP_RIP.1/TRANSIENT, FPR_UNO.1, FDP_ROL.1/FIREWALL and FDP_SDI.2 security functional requirements. The TSFs behind these are implemented by API classes. The firewall security functions FDP_ACC.2/FIREWALL and FDP_ACF.1/FIREWALL shall protect the access to private and internal data of the objects.

O.TRANSACTION
Directly met by FDP_ROL.1/FIREWALL, FDP_RIP.1/ABORT, FDP_RIP.1/ODEL, FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray, FDP_RIP.1/bArray, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL, FDP_RIP.1/TRANSIENT and FDP_RIP.1/OBJECTS (more precisely, by the element FDP_RIP.1.1/ABORT).

O.OBJ-DELETION
This security objective specifies that deletion of objects is secure. The security objective is met by the security functional requirements FDP_RIP.1/ODEL and FPT_FLS.1/ODEL.

O.DELETION
This security objective specifies that applet and package deletion must be secure. The non-introduction of security holes is ensured by the ADEL access control policy (FDP_ACC.2/ADEL, FDP_ACF.1/ADEL). The integrity and confidentiality of data that does not belong to the deleted applet or package is a by-product of this policy as well. Non-accessibility of deleted data is met by FDP_RIP.1/ADEL and the TSFs are protected against possible failures of the deletion procedures (FPT_FLS.1/ADEL, FPT_RCV.3/Installer). The security functional requirements of the class FMT (FMT_MSA.1/ADEL, FMT_MSA.3/ADEL, and FMT_SMR.1/ADEL) included in the group ADELG also contribute to meet this objective.

O.LOAD
This security objective specifies that the loading of a package into the card must be secure. Evidence of the origin of the package is enforced (FCO_NRO.2/CM) and the integrity of the corresponding data is under the control of the PACKAGE LOADING information flow policy (FDP_IFC.2/CM, FDP_IFF.1/CM) and FDP_UIT.1/CM. Appropriate identification (FIA_UID.1/CM) and transmission mechanisms are also enforced (FTP_ITC.1/CM).

O.INSTALL
This security objective specifies that installation of applets must be secure. Security attributes of installed data are under the control of the FIREWALL access control policy (FDP_ITC.2/Installer), and the TSFs are protected against possible failures of the installer (FPT_FLS.1/Installer, FPT_RCV.3/Installer).

O.SCP.IC The SCP.IC is a part of the TOE supporting TSFs of the upper layer of the TOE and more specially FPT_FLS.1/JCS and FPT_PHP.3/SCP.

O.SCP.RECOVERY
The SCP is a part of the TOE supporting TSFs of the upper layer of the TOE, especially for recovery operations as FPT_RCV.3/SCP.

O.Secure_API
The security objective is met by the following SFR FPT_FLS.1/SecureAPI, FPT_ITT.1/SecureAPI and FPR_UNO.1/SecureAPI.

O.RNG
The security objective O.RNG is met by the following SFR FCS_RNG.1.

O.JCAPI-Services
The security objective is met by the following SFR FCS_COP.1/Hash.

O.REMOTE_SERVICE_AUDIT
The security objective is met by the following SFR: FMT_MTD.1/GemActivate, FMT_SMR.1/GemActivate and FMT_SMF.1/GemActivate.

O.REMOTE_SERVICE_ACTIVATION
The security objective is met by the following SFR: FMT_SMR.1/GemActivate, FMT_SMF.1/GemActivate, FMT_MOF.1/GemActivate and FMT_MSA.1/GemActivate.

O.Secure_Load_ACode
The security objective is met by the following SFRs:

- FDP_UIT.1/CCM, FDP_ROL.1/CCM, FDP_ITC.2/CCM, FPT_FLS.1/CCM as defined for O.CARD-MANAGEMENT but applied to OS Update.
- 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) as defined for O.CARD-MANAGEMENT but applied to OS Update.
- 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) as defined for O.CARD-MANAGEMENT but applied to OS Update.
- FMT_SMR.1/GemActivate, FMT_SMF.1/GemActivate, FMT_MSA.1/GemActivate, FMT_MOF.1/GemActivate as defined for O.REMOTE_SERVICE_ACTIVATION but applied to OS Update.
- FCS_COP.1/DAP, FDP_ACC.1/GemActivate, FDP_ACF.1/GemActivate and FMT_MSA.3/GemActivate to ensure authenticity and integrity of additional software loading through the DAP mechanism.

O.Secure_AC_Activation
The security objective is met by the following SFRs:

- FDP_UIT.1/CCM, FDP_ROL.1/CCM, FDP_ITC.2/CCM, FPT_FLS.1/CCM as defined for O.CARD-MANAGEMENT but applied to OS Update.
- FMT_SMR.1/GemActivate, FMT_SMF.1/GemActivate, FMT_MSA.1/GemActivate, FMT_MOF.1/GemActivate as defined for O.REMOTE_SERVICE_ACTIVATION but applied to OS Update.

O.TOE_Identification
The security objective is met by the following SFRs: FDP_ROL.1/CCM, FDP_ITC.2/CCM as defined for O.CARD-MANAGEMENT but applied to OS Update, and FIA_ATD.1/OS-UPDATE.

O.CONFID-OS-UPDATE.LOAD
The security objective is met by the following SFR: FTP_TRP.1/OS-UPDATE.

7.4.1.3 SFR Dependency Rationale

Security Functional Requirement	CC dependencies	Satisfied dependencies
FDP_UIT.1/CCM	(FDP_ACC.1 or FDP_IFC.1) and (FTP_ITC.1 or FTP_TRP.1)	FDP_ACC.1/SD FTP_ITC.1/SC
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 **See rationale**
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

Security Functional Requirement	CC dependencies	Satisfied dependencies
		FCS_CKM.4
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	(FDP_ACF.1)	FDP_ACF.1/FIREWALL
FDP_ACF.1/FIREWALL	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.2/FIREWALL FMT_MSA.3/FIREWALL
FDP_IFC.1/JCVM	(FDP_IFF.1)	FDP_IFF.1/JCVM
FDP_IFF.1/JCVM	(FDP_IFC.1) and (FMT_MSA.3)	FDP_IFC.1/JCVM, FMT_MSA.3/JCVM
FDP_RIP.1/OBJECTS	No dependencies	
FMT_MSA.1/JCRE	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.2/FIREWALL, FMT_SMR.1 **See rationale**
FMT_MSA.1/JCVM	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.2/FIREWALL FDP_IFC.1/JCVM FMT_SMF.1 FMT_SMR.1
FMT_MSA.2/FIREWALL_JCVM	(FDP_ACC.1 or FDP_IFC.1) and (FMT_MSA.1) and (FMT_SMR.1)	FDP_ACC.2/FIREWALL FDP_IFC.1/JCVM FMT_MSA.1/JCRE FMT_MSA.1/JCVM FMT_SMR.1
FMT_MSA.3/FIREWALL	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/JCRE FMT_MSA.1/JCVM

Security Functional Requirement	CC dependencies	Satisfied dependencies
		FMT_SMR.1
FMT_MSA.3/JCVM	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/JCVM FMT_SMR.1
FMT_SMF.1	No dependencies	
FMT_SMR.1	(FIA_UID.1)	FIA_UID.2/AID
FCS_CKM.1/TDES	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_COP.1/TDES_CIPHER FCS_COP.1/TDES_MAC FCS_CKM.4
FCS_CKM.1/AES	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_COP.1/AES_CIPHER FCS_COP.1/AES_MAC FCS_CKM.4
FCS_CKM.1/RSA	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_COP.1/RSA_SIGN FCS_COP.1/RSA_CIPHER FCS_CKM.4
FCS_CKM.1/ECDSA	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_COP.1/ECDSA_SIGN FCS_CKM.4
FCS_CKM.1/HMAC	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_COP.1/HMAC FCS_CKM.4
FCS_CKM.1/ECPF	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_COP.1/ECDSA_KEY_GEN FCS_CKM.4
FCS_CKM.1/ECDH	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_COP.1/ECDH FCS_CKM.4
FCS_CKM.1/DHGen	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_COP.1/DH_KEY_GEN FCS_CKM.4
FCS_CKM.4	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2)	FCS_CKM.1/TDES FCS_CKM.1/AES FCS_CKM.1/RSA FCS_CKM.1/ECDSA FCS_CKM.1/HMAC FCS_CKM.1/ECPF FCS_CKM.1/ECDH FCS_CKM.1/DHGen
FCS_COP.1/TDES_CIPHER	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/TDES FCS_CKM.4
FCS_COP.1/TDES_MAC	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/TDES FCS_CKM.4
FCS_COP.1/AES_CIPHER	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/AES FCS_CKM.4

Security Functional Requirement	CC dependencies	Satisfied dependencies
FCS_COP.1/AES_MAC	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/AES FCS_CKM.4
FCS_COP.1/RSA_SIGN	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/RSA FCS_CKM.4
FCS_COP.1/RSA_CIPHER	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/RSA FCS_CKM.4
FCS_COP.1/ECDSA_SIGN	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/ECDSA FCS_CKM.4
FCS_COP.1/ECDH	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/ECDH FCS_CKM.4
FCS_COP.1/HMAC	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/HMAC FCS_CKM.4
FCS_COP.1/ECDSA_KEY_GEN	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/ECPF FCS_CKM.4
FCS_COP.1/DH_KEY_GEN	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/DHGen FCS_CKM.4
FDP_RIP.1/ABORT	No dependencies	
FDP_RIP.1/APDU	No dependencies	
FDP_RIP.1/GlobalArray	No dependencies	
FDP_RIP.1/bArray	No dependencies	
FDP_RIP.1/KEYS	No dependencies	
FDP_RIP.1/TRANSIENT	No dependencies	
FDP_ROL.1/FIREWALL	(FDP_ACC.1 or FDP_IFC.1)	FDP_ACC.2/FIREWALL FDP_IFC.1/JCVM
FAU_ARP.1	(FAU_SAA.1)	**See rationale**
FDP_SDI.2/DATA	No dependencies	
FPR_UNO.1	No dependencies	
FPT_FLS.1/JCS	No dependencies	
FPT_TDC.1	No dependencies	
FIA_ATD.1/AID	No dependencies	
FIA_UID.2/AID	No dependencies	
FIA_USB.1/AID	(FIA_ATD.1)	FIA_ATD.1/AID
FMT_MTD.1/JCRE	(FMT_SMF.1) and (FMT_SMR.1)	FMT_SMF.1 FMT_SMR.1
FMT_MTD.3/JCRE	(FMT_MTD.1)	FMT_MTD.1/JCRE
FDP_ITC.2/Installer	(FDP_ACC.1 or FDP_IFC.1) and (FPT_TDC.1) and (FTP_ITC.1 or FTP_TRP.1)	FDP_IFC.2/CM FTP_ITC.1/CM FPT_TDC.1
FMT_SMR.1/Installer	(FIA_UID.1)	**See rationale**

Security Functional Requirement	CC dependencies	Satisfied dependencies
FPT_FLS.1/Installer	No dependencies	
FPT_RCV.3/Installer	(AGD_OPE.1)	AGD_OPE.1
FDP_ACC.2/ADEL	(FDP_ACF.1)	FDP_ACF.1/ADEL
FDP_ACF.1/ADEL	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.2/ADEL FMT_MSA.3/ADEL
FDP_RIP.1/ADEL	No dependencies	
FMT_MSA.1/ADEL	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.2/ADEL FMT_SMF.1/ADEL FMT_SMR.1/ADEL
FMT_MSA.3/ADEL	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/ADEL FMT_SMR.1/ADEL
FMT_SMF.1/ADEL	No dependencies	
FMT_SMR.1/ADEL	(FIA_UID.1)	**See rationale**
FPT_FLS.1/ADEL	No dependencies	
FDP_RIP.1/ODEL	No dependencies	
FPT_FLS.1/ODEL	No dependencies	
FCO_NRO.2/CM	(FIA_UID.1)	FIA_UID.1/CM
FDP_IFC.2/CM	(FDP_IFF.1)	FDP_IFF.1/CM
FDP_IFF.1/CM	(FDP_IFC.1) and (FMT_MSA.3)	FDP_IFC.2/CM FMT_MSA.3/CM
FDP_UIT.1/CM	(FDP_ACC.1 or FDP_IFC.1) and (FTP_ITC.1 or FTP_TRP.1)	FDP_IFC.2/CM FTP_ITC.1/CM
FIA_UID.1/CM	No dependencies	
FMT_MSA.1/CM	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_IFC.2/CM FMT_SMF.1/CM FMT_SMR.1/CM
FMT_MSA.3/CM	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/CM FMT_SMR.1/CM
FMT_SMF.1/CM	No dependencies	
FMT_SMR.1/CM	(FIA_UID.1)	FIA_UID.1/CM
FTP_ITC.1/CM	No dependencies	
FPT_TST.1/SCP	No dependencies	
FPT_PHP.3/SCP	No dependencies	
FPT_RCV.3/SCP	(AGD_OPE.1)	AGD_OPE.1
FPT_RCV.4/SCP	No dependencies	
FCS_COP.1/Hash	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	**See rationale**
FCS_RNG.1	No dependencies	
FPT_FLS.1/SecureAPI	No dependencies	
FPT_ITT.1/SecureAPI	No dependencies	
FPR_UNO.1/SecureAPI	No dependencies	
FMT_SMR.1/GemActivate	(FIA_UID.1)	FIA_UID.1/CM

Security Functional Requirement	CC dependencies	Satisfied dependencies
		FIA_UID.2/AID FIA_UID.1/SC
FMT_SMF.1/GemActivate	No dependencies	
FMT_MOF.1/GemActivate	(FMT_SMF.1) and (FMT_SMR.1)	FMT_SMR.1/GemActivate FMT_SMF.1/GemActivate
FMT_MSA.1/GemActivate	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FMT_SMR.1/GemActivate FMT_SMF.1/GemActivate **See rationale**
FMT_MTD.1/GemActivate	(FMT_SMF.1) and (FMT_SMR.1)	FMT_SMR.1/GemActivate, FMT_SMF.1/GemActivate
FDP_ACC.1/GemActivate	(FDP_ACF.1)	FDP_ACF.1/ GemActivate
FDP_ACF.1/GemActivate	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.1/GemActivate FMT_MSA.3/GemActivate
FMT_MSA.3/GemActivate	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/GemActivate FMT_SMR.1/GemActivate
FTP_TRP.1/OS-UPDATE	No dependencies	
FIA_ATD.1/OS-UPDATE	No dependencies	

Rationale for the exclusion of dependencies:

- **The dependency FPT_TDC.1 of FDP_ITC.2/CCM is unsupported.**
  See rationale in PP.
- **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.
- **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.
- **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.
- **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.
- **The dependencies of FCS_COP.1/Hash are unsupported**
  Hash operation does not require any key.
- **The dependency FDP_ACC.1 or FDP_IFC.1 of FMT_MSA.1/GemActivate is unsupported**
  GemActivate Access Control policy is dedicated to TOE services linked to TSF data, therefore no user data is used requiring link to FDP family.

7.4.2 TOE security objectives coverage for Global Privacy Framework

7.4.2.1 TOE security objectives coverage – Mapping table

The following table provides an overview for security functional requirements coverage. It also gives an evidence for sufficiency and necessity of the chosen SFRs.
The rationale in this paragraph comes from [EAC2PP] §6.3.1

	OT.AC_Pers_EAC2	*OT.Data_Integrity*	*OT.Data_Authenticity*	*OT.Data_Confidentiality*	*OT.Identification*	*OT.Prot_Abuse-Func*	*OT.Prot_Inf_Leak*	*OT.Prot_Phys-Tamper*	*OT.Prot_Malfuntion*	*OT.Sens_Data_EAC2*
FCS_CKM.1/DH_PACE (o)		X	X	X						X
FCS_CKM.4/PACE (o)		X	X	X						X
FCS_COP.1/PACE_ENC (o)				X						X
FCS_COP.1/PACE_MAC (o)		X	X							
FCS_COP.1/PACE_CAM (o)		X	X							X
FCS_RNG.1/PACE (o)		X	X	X						X
FIA_AFL.1/PACE (o)		X	X	X						
FIA_UID.1/PACE (o)		X	X	X						X
FIA_UAU.1/PACE (o)		X	X	X						X
FIA_UAU.4/PACE (o)		X	X	X						X
FIA_UAU.5/PACE (o)		X	X	X						X
FIA_UAU.6/PACE (o)		X	X	X						X
FDP_RIP.1/PACE(p)	X	X	X	X						X
FTP_ITC.1/PACE (o)		X	X	X						X
FMT_SMF.1/PACE (o) (p)	X	X	X	X	X					X
FMT_SMR.1/PACE (o) (p)	X	X	X	X	X					X
FMT_LIM.1/PERSO (o) (p)						X				
FMT_LIM.2/PERSO (o) (p)						X				

	OT.AC_Pers_EAC2	*OT.Data_Integrity*	*OT.Data_Authenticity*	*OT.Data_Confidentiality*	*OT.Identification*	*OT.Prot_Abuse-Func*	*OT.Prot_Inf_Leak*	*OT.Prot_Phys-Tamper*	*OT.Prot_Malfuntion*	*OT.Sens_Data_EAC2*
FMT_MTD.1/INI_ENA (p)	X				X					
FMT_MTD.1/INI_DIS (p)	X				X					
FMT_MTD.1/KEY_READ (o)	X	X	X	X						X
FPT_EMS.1 (o) (p)							X			
FPT_FLS.1 (o) (p)							X		X	
FPT_TST.1 (o) (p)							X		X	
FPT_PHP.3 (o) (p)		X		X			X	X		
FCS_COP.1/SHA (o)		X	X	X						X
FCS_COP.1/SIG_VER (o)		X	X	X						X
FIA_API.1/CA (o)		X	X	X						X
FIA_UID.1/EAC2_Terminal (o)	X	X	X	X						X
FIA_UAU.1/EAC2_Terminal (o)	X	X	X	X						X
FIA_UAU.6/CA (o)		X	X	X						X
FTP_ITC.1/CA2 (o)		X	X	X						X
FMT_MTD.1/Initialize_PIN (p)	X	X	X	X						X

Table 20: Security Functional Requirement Rationale

Note: SFR followed by (o) (respectively (p)) means SFR is applicable in Operational phase (respectively (p)) personalization phase.

7.4.2.2 TOE security objectives coverage – Rationale

To achieve the security objectives of the TOE, the security functional requirements must be suitable. A detailed justification for this suitability is given below.

OT.Identification

The security objective **OT.Identification** addresses the storage of initialization and pre-personalization data in its non-volatile memory. This data includes the IC identification data that uniquely identify the TOE's chip. The SFR **FMT_MTD.1/INI_ENA** allows only the manufacturer to write initialization and pre-personalization data (including the personalization agent key). The **SFR FMT_MTD.1/INI_DIS** requires the personalization agent to disable access to initialization and pre-personalization data in the life cycle phase operational use. The SFRs **FMT_SMF.1/PACE,** and **FMT_SMR.1/PACE** support the related functions and roles.

OT.AC_Pers_EAC2
The security objective **OT.AC_Pers_EAC2** ensures that only the personalization agent can write user- and TSF-Data into the TOE, and that some of this data cannot be altered after personalization. This property is covered by terminal identification/authentication as required by the SFRs **FIA_UID.1/EAC2_Terminal** and **FIA_UAU.1/EAC2_Terminal**. The SFRs **FMT_SMF.1/PACE** and **FMT_SMR.1/PACE** support the related functions and roles. Since only an EAC2 terminal can reach the necessary authorization level, using and managing the PIN (the related SFR is **FMT_MTD.1/Initialize_PIN**) also supports the achievement of this objective. **FDP_RIP.1/PACE** requires erasing the temporal values PIN and PUK. The justification for the SFRs **FMT_MTD.1/INI_ENA** and **FMT_MTD.1/INI_DIS** arises from the justification for OT.Identification above with respect to the pre-personalization data. Finally, **FMT_MTD.1/KEY_READ** ensures that cryptographic keys for EAC2 cannot be read by users.

OT.Data_Integrity
The security objective **OT.Data_Integrity** ensures that the TOE always ensures integrity of stored user- and TSF-Data and, after Terminal- and Chip Authentication 2, of these data exchanged (physical manipulation and unauthorized modifying). Physical manipulation is addressed by **FPT_PHP.3**. The Personalization Agent must identify and authenticate themselves according to **FIA_UID.1/PACE** and **FIA_UAU.1/PACE** before accessing these data. **FIA_UAU.4/PACE, FIA_UAU.5/PACE and FCS_CKM.4/PACE** represent some required specific properties of the protocols used. The SFR **FMT_SMR.1/PACE** manages the roles and the **SFR FMT_SMF.1/PACE** manages the TSF management functions.
Unauthorized modifying of the exchanged data is addressed, in the first line, by **FTP_ITC.1/PACE** using **FCS_COP.1/PACE_MAC, FCS_COP.1/PACE_CAM**. For PACE secured data exchange, a prerequisite for establishing this trusted channel is a successful PACE Authentication (**FIA_UID.1/PACE, FIA_UAU.1/PACE**) using **FCS_CKM.1/DH_PACE** and possessing the special properties **FIA_UAU.5/PACE, FIA_UAU.6/PACE**.
FIA_AFL.1/PACE allows to manage errors in secure channel management.
FDP_RIP.1/PACE requires erasing the values of session keys (here: for KMAC).
The session keys are destroyed according to **FCS_CKM.4/PACE** after use.
A specific authorization level is achieved by terminal identification/ authentication as required by the SFRs **FIA_UID.1/EAC2_Terminal**, **FIA_UAU.1/EAC2_Terminal**, supported by **FCS_COP.1/SIG_VER**. The TA2 protocol uses the result of PACE authentication (**FIA_UID.1/PACE**, **FIA_UAU.1/PACE**) being, in turn, supported by **FCS_CKM.1/DH_PACE**. Since PACE can use the PIN as the shared secret, using and management of PIN (**FMT_MTD.1/Initialize_PIN)** also support achievement of this objective. Unauthorized modifying of the exchanged data is addressed by **FTP_ITC.1/CA2** and **FTP_ITC.1/PACE** using **FCS_COP.1/PACE_MAC**. A prerequisite for establishing this trusted channel is a successful Chip Authentication 2, cf. **FIA_API.1/CA** using **FCS_CKM.1/DH_PACE** possessing the special properties **FIA_UAU.5/PACE** and **FIA_UAU.6/CA**. The SFRs **FCS_COP.1/SHA** and **FCS_RNG.1/PACE** represent a general support for cryptographic operations needed.
The SFR **FMT_MTD.1/KEY_READ** requires that data cannot be unauthorized read afterwards.

OT.Data_Authenticity
The security objective **OT.Data_Authenticity** aims ensuring authenticity of the User and TSF data (after the PACE authentication - after Terminal and the Chip Authentication 2) by enabling its verification on both the terminal-side and by an active verification by the TOE itself. This objective is mainly achieved by **FTP_ITC.1/CA2** and **FTP_ITC.1/PACE** using **FCS_COP.1/PACE_MAC, FCS_COP.1/PACE_CAM**. A prerequisite for establishing this trusted channel is a successful PACE or Chip and Terminal Authentication v.1 or v.2 (**FIA_UID.1/PACE**, **FIA_UAU.1/PACE**, **FIA_API.1/CA**) using **FCS_CKM.1/DH_PACE** and possessing the special properties **FIA_UAU.5/PACE**, **FIA_UAU.6/PACE, FIA_UAU.6/CA**. **FDP_RIP.1/PACE** requires erasing the values of session keys (here: for KMAC). A prerequisite for successful CA2 is an accomplished TA2 as required by **FIA_UID.1/EAC2_Terminal**, **FIA_UAU.1/EAC2_Terminal**, supported by **FCS_COP.1/SIG_VER**. Since PACE can use the PIN as the shared secret, the use of the PIN

(**MT_MTD.1/Initialize_PIN**) also supports achieving this objective. **FDP_RIP.1/PACE** requires erasing the temporal values of the PIN and PUK.
FIA_UAU.4/PACE, FIA_UAU.5/PACE and **FCS_CKM.4/PACE** represent some required specific properties of the protocols used. The SFRs **FCS_COP.1/SHA** and **FCS_RNG.1/PACE** represent the general required support for cryptographic operations. **FIA_AFL.1/PACE** allows to manage errors in secure channel management. The SFR **FMT_MTD.1./KEY_READ** restricts the access to the PACE passwords.
The SFR **FMT_SMR.1/PACE** manages the roles and the SFR **FMT_SMF.1/PACE** manages the TSF management functions.

OT.Data_Confidentiality
The security objective **OT.Data_Confidentiality** ensures that the TOE always ensures confidentiality of the user- and TSF-Data stored and, after Terminal- and Chip Authentication 2, of their exchange. Physical manipulation is addressed by **FPT_PHP.3**.
A specific authorization level is achieved by terminal identification/authentication as required by the SFRs **FIA_UID.1/EAC2_Terminal**, **FIA_UAU.1/EAC2_Terminal**, supported by **FCS_COP.1/SIG_VER**. The TA2 protocol uses the result of the PACE authentication (**FIA_UID.1/PACE**, **FIA_UAU.1/PACE,** confidentiality of the PACE passwords is ensured by **FMT_MTD.1/KEY_READ**) being, in turn, supported by **FCS_CKM.1/DH_PACE**. Since PACE can use the PIN as the shared secret, the SFR **MT_MTD.1/Initialize_PIN** also supports to achieve this objective. **FDP_RIP.1** requires erasing the temporal values of the PIN and PUK.
FIA_UAU.4/PACE, FIA_UAU.5/PACE and **FCS_CKM.4/PACE** represent some required specific properties of the protocols used. This objective for the data exchanged is mainly achieved by **FTP_ITC.1/CA2** and **FTP_ITC.1/PACE** using **FCS_COP.1/PACE_ENC**.
A prerequisite for establishing this trusted channel is a successful PACE or Chip Authentication v.1 or v.2 (**FIA_UID.1/PACE**, **FIA_UAU.1/PACE**, **FIA_API.1/CA**) using **FCS_CKM.1/DH_PACE** and possessing the special properties **FIA_UAU.5/PACE**, **FIA_UAU.6/PACE, FIA_UAU.6/CA**.
FDP_RIP.1/PACE requires erasing the values of session keys (here: for KENC). **FIA_AFL.1/PACE** allows to manage errors in PACE secure channel management. The **SFR FMT_MTD.1./KEY_READ** restricts the access to the PACE passwords.
The SFRs **FCS_COP.1/SHA** and **FCS_RNG.1/PACE** represent the general support for cryptographic operations.
The SFRs **FMT_SMR.1/PACE** manages the roles and the SFRs **FMT_SMF.1/PACE** manages the TSF management functions.

OT.Prot_Abuse_Func
The security objective **OT.Prot_Abuse_Func** “Protection against Abuse of Functionality” is ensured by the SFR **FMT_LIM.1/PERSO** and **FMT_LIM.2/PERSO** which prevent misuse of test functionality of the TOE or other features which may not be used after TOE Delivery.

OT.Prot_Inf_Leak
OT.Prot_Inf_Leak “Protection against Information Leakage” requires the TOE to protect confidential TSF data stored and/or processed in the travel document’s chip against disclosure,
- by measurement and analysis of the shape and amplitude of signals or the time between events found by measuring signals on the electromagnetic field, power consumption, clock, or I/O lines which is addressed by the **SFR FPT_EMS.1**,
- by forcing a malfunction of the TOE which is addressed by the **SFR FPT_FLS.1** and **FPT_TST.1**, and/or
- by a physical manipulation of the TOE which is addressed by the **SFR FPT_PHP.3**.

OT.Prot_Phys_Tamper
The security objective **OT.Prot_Phys_Tamper** “Protection against Physical Tampering” is covered by the SFR **FPT_PHP.3**.

OT.Prot_Malfunction
The security objective **OT.Prot_Malfunction** “Protection against Malfunctions” is covered by (i) the SFR **FPT_TST.1** which requires self-tests to demonstrate the correct operation and tests of authorized users to verify the integrity of TSF data and TSF code, and (ii) the SFR **FPT_FLS.1** which requires a secure state in case of detected failure or operating conditions possibly causing a malfunction.

OT.Sens_Data_EAC2

The security objective of **OT.Sens_Data_EAC2** aims to explicitly protect sensitive (as opposed to common) user and TSF-Data. A specific authorization level is achieved by terminal identification/authentication as required by the SFRs **FIA_UID.1/EAC2_Terminal**, **FIA_UAU.1/EAC2_Terminal**, supported by **FCS_COP.1/SIG_VER**. The TA2 protocol uses the result of the PACE authentication (**FIA_UID.1/PACE**, **FIA_UAU.1/PACE,** confidentiality of the PACE passwords is ensured by **FMT_MTD.1/KEY_READ**) being, in turn, supported by **FCS_CKM.1/DH_PACE**. Since PACE can use the PIN as the shared secret, the PIN (**FMT_MTD.1/Initialize_PIN**) also supports to achieve this objective. **FDP_RIP.1** requires erasing the temporal values of the PIN and PUK.

FIA_UAU.4/PACE, **FIA_UAU.5/PACE, FIA_UAU.6/PACE** and **FCS_CKM.4** represent some specific properties of the used protocols. The objective for the data exchanged is mainly achieved by **FTP_ITC.1/CA2** and **FTP_ITC.1/PACE** using **FCS_COP.1/PACE_ENC**. A prerequisite for establishing this trusted channel is a successful Chip Authentication 2, cf. **FIA_API.1/CA** using **FCS_CKM.1/DH_PACE** and possessing the special properties **FIA_UAU.5/PACE**, and **FIA_UAU.6/CA**. As a prerequisite of this trusted channel, a trusted channel is established with the PACE protocol using **FIA_UID.1/PACE, FIA_UAU.1/PACE** and **FCS_CKM.1/DH_PACE** and possessing the special properties **FIA_UAU.5/PACE, FIA_UAU.6/PACE**.

CA2 provides an evidence of possessing the Chip Authentication Private Key (SK_{PICC}). **FMT_MTD.1/KEY_READ** requires making this key unreadable by users. Thus its value remains confidential. **FDP_RIP.1** requires erasing the values of SK_{PICC} and session keys, here for K_{ENC}. The SFRs **FCS_COP.1/SHA** and **FCS_RNG.1** represent the general required support for cryptographic operations.
The SFR **FMT_SMR.1/PACE** manages the roles and the SFR **FMT_SMF.1/PACE** manages the TSF management functions.

7.4.2.3 SFR Dependency Rationale

The dependency analysis for the security functional requirements shows that the basis for mutual support and internal consistency between all defined functional requirements is satisfied. All dependencies between the chosen functional components are analyzed, and non-dissolved dependencies are appropriately explained.
The dependency analysis has directly been made within the description of each SFR in section 7.2.2.4 above. All dependencies, being expected by [CC2] and by extended component definitions, are either fulfilled or their non-fulfillment is justified.

7.4.3 SAR Dependency Rationale

Security Assurance Requirement	CC dependencies	Satisfied dependencies
ADV_ARC.1	(ADV_FSP.1) and (ADV_TDS.1)	ADV_FSP.4 ADV_TDS.3
ADV_FSP.4	(ADV_TDS.1)	ADV_TDS.3
ADV_TDS.3	(ADV_FSP.4)	ADV_FSP.4
ADV_IMP.1	(ADV_TDS.3) and (ALC_TAT.1)	ADV_TDS.3 ALC_TAT.1
AGD_OPE.1	(ADV_FSP.1)	ADV_FSP.4
AGD_PRE.1	No dependencies	
ALC_CMC.4	(ALC_CMS.1) and (ALC_DVS.1) and (ALC_LCD.1)	ALC_CMS.4

Security Assurance Requirement	CC dependencies	Satisfied dependencies
		ALC_DVS.2 ALC_LCD.1
ALC_CMS.4	No dependencies	
ALC_DEL.1	No dependencies	
ALC_DVS.2	No dependencies	
ALC_LCD.1	No dependencies	
ALC_TAT.1	(ADV_IMP.1)	ADV_IMP.1
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.4 ASE_INT.1 ASE_REQ.2
ATE_COV.2	(ADV_FSP.2) and (ATE_FUN.1)	ADV_FSP.4 ATE_FUN.1
ATE_DPT.1	(ADV_ARC.1) and (ADV_TDS.2) and (ATE_FUN.1)	ADV_ARC.1 ADV_TDS.3 ATE_FUN.1
ATE_FUN.1	(ATE_COV.1)	ATE_COV.2
ATE_IND.2	(ADV_FSP.2) and (AGD_OPE.1) and (AGD_PRE.1) and (ATE_COV.1) and (ATE_FUN.1)	ADV_FSP.4 AGD_OPE.1 AGD_PRE.1 ATE_COV.2 ATE_FUN.1
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.1)	ADV_ARC.1 ADV_FSP.4 ADV_IMP.1 ADV_TDS.3 AGD_OPE.1 AGD_PRE.1 ATE_DPT.1

The table here-above shows that all SAR dependencies are met.

7.4.4 Rationale for the Security Assurance Requirements

EAL4 is required for this type of TOE and product since it is intended to defend against sophisticated attacks. This evaluation assurance level allows a developer to gain maximum assurance from positive security engineering based on good practices. EAL4 represents the highest practical level of assurance expected for a commercial grade product. In order to provide a meaningful level of assurance that the TOE and its embedding product provide an adequate level of defense against such attacks: the evaluators should have access to the low level design and source code. The lowest for which such access is required is EAL4.

- ALC_DVS.2 SUFFICIENCY OF SECURITY MEASURES

Development security is concerned with physical, procedural, personnel and other technical measures that may be used in the development environment to protect the TOE and the embedding product. The standard ALC_DVS.1 requirement mandated by EAL4 is not enough. Due to the nature of the TOE and embedding product, it is necessary to justify the sufficiency of these procedures to protect their confidentiality and integrity. ALC_DVS.2 has no dependencies.

- AVA_VAN.5 ADVANCED METHODICAL VULNERABILITY ANALYSIS

The TOE is intended to operate in hostile environments. AVA_VAN.5 "Advanced methodical vulnerability analysis" is considered as the expected level for Java Card technology-based products hosting sensitive applications, in particular in payment and identity areas. AVA_VAN.5 has dependencies on ADV_ARC.1, ADV_FSP.1, ADV_TDS.3, ADV_IMP.1, AGD_PRE.1 and AGD_OPE.1. All of them are satisfied by EAL4.

7.5 COMPOSITION TASKS – SFR PART

The following table (see next page) lists the SFRs that are declared in the security target [ST-IC], and separates them in 3 groups, as requested in [CCDB]:

- **RP-SFR**: Irrelevant Platform-SFRs not being used by the Composite-ST
- **RP-SFR-SERV**: Relevant Platform-SFRs being used by the Composite-ST to implement a security service with associated TSFI
- **RP-SFR-MECH**: Relevant Platform-SFRs being used by the Composite-ST because of its security properties providing protection against attacks to the TOE as a whole and are addressed in ADV_ARC. These required security properties are a result of the security mechanisms and services that are implemented in the Platform TOE.

The link between the relevant platform-SFRs and the composite product SFRs is described below:

SFRs relative to Bootloader:

- FMT_LIM.1 (LOADER), FMT_LIM.2 (LOADER), FDP_ACC.1 (LOADER), FDP_ACF.1 (LOADER): No direct link to the composite product SFRs, since the IC Loader is no more available after phase 5. However, these IC TOE SFRs are essential to protect the composite TOE during phases 4, 5 (covered by the ALC assurance classes).
- FTP_ITC.1, FDP_UCT.1, FDP_UIT.1 and FIA_API.1 are linked to the secure communication when using the Bootloader. No direct link to the composite product SFRs

FAU_SAS.1: No link to TOE SFRs but used for the composite-product identification.

FDP_SDC.1: Link to TOE SFRs FPT_RCV.4/SCP and FPT_TST.1/SCP

FDP_SDI.2: Link to TOE SFR FDP_SDI.2

FRU_FLT.2, FPT_FLS.1: Link to TOE SFRs FPT_RCV.3/SCP, FPT_RCV.4/SCP, FPT_FLS.1/JCS, FPT_FLS.1, FPT_TST.1

FMT_LIM.1, FMT_LIM.2: Link to TOE SFRs FAU_ARP.1, FPT_FLS.1/JCS, FMT_LIM.1/PERSO, FMT_LIM.2/PERSO

FPT_PHP.3: Link to TOE SFRs FAU_ARP.1, FPT_RCV.3/SCP, FPT_RCV.4/SCP, and FPT_PHP.3

FDP_IFC.1, FDP_ITT.1, FPT_ITT.1: Link to TOE SFR FPR_UNO.1 and FPT_EMS.1

FCS_RNG.1/PTG.2: Link to TOE SFR FCS_RNG.1 and FCS_RNG.1/PACE

FCS_RNG.1/RGS-IC: No link to TOE SFR

FCS_COP.1/TDES: Link to TOE SFRs FCS_COP.1/TDES_CIPHER and FCS_COP.1/TDES_MAC, FCS_CKM.1/TDES, FCS_CKM.A/DH_PACE.

FCS_COP.1/AES: Link to TOE SFRs FCS_COP.1/AES_CIPHER and FCS_COP.1/AES_MAC, FCS_CKM.1/AES, FCS_RNG.1, FCS_CKM.1/DH_PACE, FCS_COP.1/PACE_ENC, FCS_COP.1/PACE_MAC, FCS_RNG.1/PACE

FDP_ACC.2 (MEMORIES): Link to TOE SFR FDP_ACC.2/FIREWALL

FDP_ACF.1 (MEMORIES): Link to TOE SFR FDP_ACF.1/FIREWALL

FMT_MSA.1 (MEMORIES): Link to TOE SFRs FMT_MSA.1/JCRE and FMT_MSA.1/JCVM

FMT_MSA.3 (MEMORIES): Link to TOE SFR FMT_MSA.3/FIREWALL

FMT_SMF.1 (MEMORIES): Link to TOE SFR FMT_SMF.1

Other SFRs from the TOE are not directly supported by SFRs of [ST-IC].
We can therefore conclude that the SFRs of the current ST and [ST-IC] are consistent.

Platform-SFR	Platform-SFR content	RP_SFR	RP_SFR-SERV	RP_SFR-MECH
FRU_FLT.2	**Limited fault tolerance**: The TSF shall ensure the operation of all the TOE's capabilities when the following failures occur: exposure to operating conditions which are not detected according to the requirement Failure with preservation of secure state (FPT_FLS.1).			**X**
FPT_FLS.1	**Failure with preservation of secure state**: The TSF shall preserve a secure state when the following types of failures occur: exposure to operating conditions which may not be tolerated according to the requirement Limited fault tolerance (FRU_FLT.2) and where therefore a malfunction could occur.			**X**
FAU_SAS.1	**Audit storage:** The TSF shall provide the test process before TOE Delivery with the capability to store the Initialization Data and/or Pre-personalization Data and/or supplements of the Security IC Embedded Software in the NVM.			**X**
FDP_SDC.1	**Stored data confidentiality:** The TSF shall ensure the confidentiality of the information of the user data while it is stored in the FLASH, RAM or ROM		**X**	
FDP_SDI.2	**Stored data integrity monitoring and action:** The TSF shall monitor user data stored in containers controlled by the TSF for error on all objects. Upon detection of a data integrity error, the TSF shall enforce a device RESET or an interrupt		**X**	
FMT_LIM.1	**Limited capabilities**: The TSF shall be designed and implemented in a manner that limits their capabilities so that in conjunction with "Limited availability (FMT_LIM.2)" the following policy is enforced: Limited capability and availability Policy (TEST).			**X**
FMT_LIM.2	**Limited availability**: The TSF shall be designed and implemented in a manner that limits their availability so that in conjunction with "Limited capabilities (FMT_LIM.1)" the following policy is enforced: Limited capability and availability Policy (TEST).			**X**

Platform-SFR	Platform-SFR content	RP_SFR	RP_SFR-SERV	RP_SFR-MECH
FPT_PHP.3	The TSF shall resist physical manipulation and physical probing to the TSF by responding automatically such that the SFRs are always enforced.			**X**
FDP_IFC.1	The TSF shall enforce the Data Processing Policy on all confidential data when they are processed or transferred by the TSF or by the Security IC Embedded Software.			**X**
FDP_ITT.1	The TSF shall enforce the Data Processing Policy to prevent the disclosure of user data when it is transmitted between physically-separated parts of the TOE.			**X**
FPT_ITT.1	The TSF shall protect TSF data from disclosure when it is transmitted between separate parts of the TOE. The different memories, the CPU and other functional units of the TOE (e.g. a cryptographic co-processor) are seen as separated parts of the TOE.			**X**
FCS_RNG.1/PTG.2	The TSF shall provide a physical true random number generator that implements: * A total failure test […] The TSF shall provide octets of bits that meet […]		**X**	
FCS_RNG.1/RGS-IC	The TSF shall provide a physical random number generator that implements: * A total failure test […] The TSF shall provide octets of bits that meet […]	**X**		
FDP_ACC.1 (MEMORIES)	The TSF shall enforce the Memory Access Control Policy on all subjects (software with privilege mode and user mode), all objects (data including code stored in memories) and all the operations defined in the Memory Access Control Policy.			**X**
FDP_ACF.1 (MEMORIES)	The TSF shall enforce the Dynamic Memory Access Control Policy to objects based on the following: software mode, the object location, the operation to be performed, and the current set of access rights. The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: the operation is allowed if and only if the software mode, the object location and the operation matches an entry in the current set of access rights.			**X**

Platform-SFR	Platform-SFR content	RP_SFR	RP_SFR-SERV	RP_SFR-MECH
	The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: none. The TSF shall explicitly deny access of subjects to objects based on the following additional rules: in Admin or User configuration, any access (read, write, execute) to the OST ROM is denied, and in User configuration, any write access to the ST NVM is denied.			
FMT_MSA.1 (MEMORIES)	The TSF shall enforce the Dynamic Memory Access Control Policy to restrict the ability to modify the security attributes current set of access rights to software running in privileged mode.			**X**
FMT_MSA.3 (MEMORIES)	The TSF shall enforce the Dynamic Memory Access Control Policy to provide minimally protective default values for security attributes that are used to enforce the SFP. The TSF shall allow none to specify alternative initial values to override the default values when an object or information is created.			**X**
FMT_SMF.1 (MEMORIES)	The TSF will be able to perform the following management functions: modification of the current set of access rights security attributes by software running in privileged mode, supporting the Dynamic Memory Access Control Policy.			**X**
FMT_LIM.1 (LOADER)	The TSF shall be designed and implemented in a manner that limits their capabilities so that in conjunction with “Limited availability (FMT_LIM.2)” the following policy is enforced: Deploying Loader functionality after locking the chip to FLASH booting mode does not allow stored user data to be disclosed or manipulated by unauthorized user.			**X**
FMT_LIM.2 (LOADER)	The TSF shall be designed and implemented in a manner that limits their availability so that in conjunction with “Limited capabilities (FMT_LIM.1)” the following policy is enforced: The TSF prevents deploying the Loader functionality after locking the chip to FLASH booting mode			**X**
FDP_ACC.1 (LOADER)	The TSF shall enforce the Loading Access Control Policy on the execution of the Standard Loader instructions and/or the Advanced Loader instructions.			**X**
FDP_ACF.1 (LOADER)	The TSF shall enforce the Loading Access Control Policy to objects based on the following: an external process may execute the Standard Loader instructions and/or the Advanced			**X**

Platform-SFR	Platform-SFR content	RP_SFR	RP_SFR-SERV	RP_SFR-MECH
	Loader instructions, depending on the presentation of valid passwords. The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: the Standard Loader instructions and/or Advanced Loader instructions can be executed only if valid passwords have been presented. The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: none. The TSF shall explicitly deny access of subjects to objects based on the following additional rules: none.			
FCS_COP.1/TDES	- Cryptographic operations to be performed in accordance with a specified algorithm and with a cryptographic key of specified sizes. - TDES with 112bit or 168bit key size		**X**	
FCS_COP.1/AES	- Cryptographic operations to be performed in accordance with a specified algorithm and with a cryptographic key of specified sizes. - AES with 128bit, 192bit and 256bit key size		**X**	
FTP_ITC.1	Inter-TSF trusted channel	**X**		
FDP_UCT.1	Basic data exchange confidentiality	**X**		
FDP_UIT.1	Basic data exchange integrity	**X**		
FIA_API.1	Authentication Proof of Identity	**X**		

8 TOE SUMMARY SPECIFICATION

8.1 UPTEQ NFC422 V1.0 JCS

GP.CardContentManagement
This security function provides the capability and a dedicated flow control for the loading, installation, extradition, registry update, selection and removal of card content and especially executable files and application instances. Such features are offered to the Card Issuer and its business partners, allowing the Card Issuer to delegate card content management to an Application Provider according to privileges assigned to the various security domains on the card. It supports delegated management and it can use DAP or Mandated DAP verification and generation of Reception token. It also checks that only the card management commands specified and allowed at each state of the smart card's life cycle are accepted, and ill-formed ones are rejected with an appropriate error response.

GP.SecurityDomain
This security function provides security domain management, as SD creation, SD selection, SD privileges setting and SD deletion in SD hierarchy. It provides means to associate or extradite an application to a security domain in order to provide services (as secure channel) to the dedicated application without sharing the related keys stored in SD. It also provides Keyset Management in SD, with Key Set creation, Key set deletion, key importation, replacement, or deletion in Key Set.
Security Domains are privileged Applications as defined in [GP23], holding cryptographic keys to be used to support Secure Channel Protocol operations and/or to authorize card content management functions. There are different types of security domain with dedicated privileges and associated operations: ISD Security domain, Supplementary Security domains, and Controlling Authority Security domains.
ISD Security domain as defined in [GP23], is the mandatory Security Domain, implicitly selected if the Application implicitly selectable on the same logical channel of the same card I/O interface is removed. It inherits of the Final Application privilege if the Application with that privilege is removed.
Supplementary Security Domains are privileged Applications with dedicated privileges:

- Token Verification Privilege
- Delegated Management Privilege
- Global Delete Privilege
- Global Lock Privilege
- Receipt Generation Privilege

Controlling Authority Security Domain is a supplementary Security Domain dedicated to the Controlling Authority with dedicated privileges. It contains Security Domains cryptographic keys needed to confidentially personalize an initial set of Secure Channel Keys of an APSD.

GP.ISD
This security function manages the Issuer Security Domain with associated functions and dedicated privileges.

GP.CASD
This security function manages supplementary Controlling Authority Security Domain with associated functions to confidential Card Content Management.

GP.VASD
This security function manages supplementary Verification Authority Security Domain with associated functions to Mandated DAP.

GP.SSD
This security function manages supplementary Security Domains with associated functions and dedicated privileges.

GP.SCP

This security function manages Secure Channel protocol according to [GP23] annex E, [GP23 Amend A], [GP23 Amend D] and [GP23 Amend F].

GP.SecureChannel
This security function provides a secure communication channel between a card and an off-card entity during an Application Session according to [GP23]. It provides an APDU flow control using the Command security level check according to Card Life cycle and type of APDU.
A Secure Channel Session is divided into three sequential phases:

- Secure Channel Initiation when the on-card Application and the off-card entity have exchanged sufficient information enabling them to perform the required cryptographic functions. The Secure Channel Session initiation always includes (at least) the authentication of the off-card entity by the on-card Application; performing also the setting of the Command security level used for the session.
- Secure Channel Operation when the on-card Application and the off-card entity exchange data within the cryptographic protection of the Secure Channel Session. The Secure Channel services offered may vary from one Secure Channel Protocol to the other;
- Secure Channel Termination when either the on-card Application or the off-card entity determines that no further communication is required or allowed via an established Secure Channel Session.

The following services are provided by the Secure Channel as defined in [GP_23] §4.3.2 and §10 using SCP03 or SCP11:

- Entity authentication in which the card or the off-card entity proves its authenticity to the other entity through a cryptographic exchange, based on session key generation and a dedicated flow control;
- Integrity and authentication in which the receiving entity (the card or off-card entity) ensures that the data being received from the sending entity (respectively the off-card entity or card) actually came from an authenticated entity in the correct sequence and has not been altered;
- Confidentiality in which data being transmitted from the sending entity (the off-card entity or card) to the receiving entity (respectively the card or off-card entity) is not viewable by an unauthenticated entity.

Note: Compared to GP.SCP which defines Secure Channel Protocols (like SCP02, SCP03…), GP.SecureChannel defines secure channel generic features and operations.

GP.GPRegistry
This security function provides access to the GlobalPlatform Registry used for:

- Store card management information;
- Store relevant application management information (e.g., AID, associated Security Domain and Privileges);
- Support card resource management data;
- Store Application Life Cycle information;
- Store card Life Cycle information;
- Track any counters associated with logs.

The content of the GlobalPlatform Registry may be accessed by administrative commands or by applet using a dedicated GP_API.

GP.OS-UPDATE_CODE_ID
This security function manages additional code identification by updating a TAG containing a patch identifier. It is associated to Platform identifier to provide the complete TOE identification.

JCS.APDUBuffer
The security function maintains a byte array buffer accessible from any applet context. This buffer is used to transfer incoming APDU header and data bytes as well as outgoing data according to [JCAPI305]. The APDU class API is designed to be transport protocol independent T=0, T=1, T=CL (as defined in ISO 7816-3).

Application note: ADPU buffer is a JCRE temporary entry point object where no associated reference can be stored in a variable or an array component.

JCS.ByteCodeExecution
This security function handles applet bytecode execution according to the rules defined in [JCVM305]. The JCVM execution may be summarized in JCVM interpreter start-up, bytecode execution and JCVM interpreter loop. The applet bytecode execution consists in:

- fetching the next bytecode to execute according to the applet's code flow control,
- decoding the next bytecode,
- executing the fetched bytecode.

The JCVM manages 5 types of objects: persistent objects, transient objects, persistent arrays (boolean, byte, short, int or reference), transient arrays (boolean, byte, short, int or reference) and static field images. For each type of object, different types of control are performed.

JCS.Firewall
This security function enforces the Firewall access control policy and the JCVM information flow control policy at runtime. It defines how accessing the following items: Static Class Fields, Array Objects, Class Instance Object Fields, Class Instance Object Methods, Standard Interface Methods, Shareable Interface Methods, Classes, Standard Interfaces, Shareable Interfaces, Array Object Methods.
Based on security attributes (Sharing, Context, Lifetime), it performs access control to object fields between objects and throws security exception when access is denied. Thus, it enforces applet isolation located in different packages and controls the access to global data containers shared by all applet instances.
The JCRE shall allocate and manage a context for each Java API package containing applets. The JCRE maintains for its own context a special system privilege so that it can perform operations that are denied to contexts of applets.

JCS.Package
This security function manages packages. A package is a structural item defined for naming, loading, storing, execution context definition. There are rules for package identification, for structure check and access rules definition. If inconsistent items are found during checks, an error message is sent.

JCS.Crypto
The security function offers the following services to applets thanks to the JavaCard API:

- Generation of random number as defined in [JCAPI305] and conformant to [AIS31] test procedure A, to be used for key values or challenges during external exchanges,
- Ciphering and deciphering operation using TDES algorithm as defined in [JCAPI305],
- Generation of 4-byte or 8-byte MAC using TDES (112 or 168 bits key) algorithm according to [JCAPI305],
- Ciphering and deciphering operation using AES (128, 192 or 256 bits key) algorithm as defined in [JCAPI305],
- Generation of 16-byte, 24-byte or 32-byte MAC using AES algorithm in CBC mode with padding scheme (NOPAD), as defined in [JCAPI305],
- Data Hash operation for message digests using SHA algorithms (SHA1, SHA224, SHA-256, SHA-384, SHA-512), as defined in [JCAPI305],
- HMAC generation based on SHA1, SHA256, SHA384 and SHA512 hash algorithms, as defined in [JCAPI305],
- Ciphering and deciphering operation using RSA with Standard and CRT algorithms, with padding scheme PKCS#1 or NOPAD, according to [JCAPI305],
- Generation and verification of RSA signatures in Standard or CRT modes, as defined in [JCAPI305]
- Generation and verification of ECDSA signatures, as defined in [JCAPI305]
- Generation of shared secrets according to the ECDH algorithm, as defined in [JCAPI305].

These operations are performed in a way to avoid revealing the key values. If the applet specifies an algorithm that the platform does not support, the JCRE refuses to perform the cryptographic operation and generates an

exception. Even if [JCAPI305] specifies some other algorithms or parameters for cryptographic operations, the use of these other values are not advised; and clearly out of scope of the TOE. See [AGD] for details.

JCS.RNG
This security function provides random value using a given algorithm with or without a seed as defined in [JCAPI305].

JCS.KeyManagement
This security function enforces key management for the different associated operations: key building, key agreement, key generation, key importation, key exportation, key masking, key destruction using standard API defined in [JCAPI305].
Key generation supports the generation of RSA and ECDSA key pairs using a secure random number generator compliant with [AIS31] test procedure A.
Key importation and exportation is done using method protecting confidentiality and integrity of key.
Key agreement enables an applet to agree on a shared secret with the external, with a method conformant to [JCAPI305]. It is built to avoid disclosure of this secret to third parties observing the exchange done for key agreement.
Key masking protects the confidentiality of cryptographic keys from being read out from the memory. It ensures the service of accessing and modifying them.
Key destruction disables the use of a key both logically and physically. Reuse is only possible after erase.

JCS.OwnerPIN
This security function supplies to applets a means to perform user identification and authentication with the OwnerPin class conformant to [JCAPI305].
It offers to create a PIN and store it securely in the persistent memory. It allows access to PIN value only to perform a secure comparison between a PIN stored in the persistent memory and a data received as parameter.
A method returns a positive result if a valid Pin has been presented during current session. If the PIN is not blocked and the comparison is successful, the validated flag is set to and the try counter is set to its maximum, otherwise the authentication fails and the associated try counter is decremented. When the validated flag is set, it is assumed that the user is authenticated.
When the try counter reaches zero, the PIN is blocked and the authentication is no more possible until the PIN is unblocked.

JCS.EraseResidualData
This security function ensures that sensitive data are locked upon the following operations as defined in [JCRE305]:

- Deletion of package and/or applications,
- Deletion of objects.

They are erased when space needs to be reused for allocation of new objects.
This security function also ensures that the sensitive temporary buffers (transient object, bArray object, APDU buffer, Cryptographic buffer) are securely cleared after their usage with respect to their life-cycle and interface as defined in [JCRE305], transient object at reset or allocation and persistent object are erased at allocation for new object.

JCS.OutOfLifeDataUndisclosure
This security function ensures that sensitive data are locked until postponed erasure on the following operations: Deletion of persistent and transient objects according to [JCRE305].

JCS.RunTimeExecution
This security function provides a secure run time environment and deals with:

- Instance registration or deletion,
- Application selection,
- Applet opcode execution,
- JCAPI methods execution,
- Logical channel management,
- APDU flow control, dispatch and buffer management,

- JCRE memory and context management,
- JCRE reference deletion,
- JCRE access rights,
- JCRE throw exception,
- JCRE security reaction.

JCS.Exception
This security function manages throwing of an instance of Exception class in the following cases:

- a SecurityException when an illegal access to an object is detected,
- a SystemException with an error code describing the error condition,
- a CryptoException in case of algorithm error or illegal use,
- any exception decided by the applet or the JCRE handled as temporary JCRE entry point object with associated JC API. It also offers a means to applet to handle exception and to JCRE to handle uncaught exception by applets.

SA.FlowControl *(Secure API)*
This security function provides means to applications to control execution flow, to detect any failure and to react if required.

SA.SecureOperation *(Secure API)*
This security function provides means to applications to securely perform data transfer and comparison, to detect any failure during operation and to react if required.

SA.RandomDelay *(Secure API)*
This security function provides means to introduce dummy operations leading to unobservability of sensitive operation.

GA.OptionalServiceActivation *(GemActivate)*
Activation is only possible for deactivated services defined in registry. Activation is done by changing internal state of optional platform service: cryptographic algorithm, package, applet instance, scalability (extension of available NVM) and NFC interface (SWP, HCI gate). It also allows activation of patch loaded on the platform. The command is available only for GemActivate under control of GemActivate Administrator. GemActivate is accessible only using a secure channel under OEM control.

GA.PatchManagement *(GemActivate)*
It manages patch identification when loaded. It allows activation of patch loaded on the platform and deactivation of activated patch by changing patch internal state. The command is available only for GemActivate under control of GemActivate Administrator. GemActivate is accessible only using a secure channel under OEM control.

GA.ServiceAudit (GemActivate)
It allows OEM or GemActivate administrator audit of actual state (deactivated, activated, inhibited) of each optional platform service described in platform registry.

GA.GemActivateActivation *(GemActivate)*
An application or a patch can be activated by GemActivate Administrator only if the following conditions are fulfilled:

- if patchability activation status is set,
- if activation command is consistent,
- if ratification counter limit is not reached,
- if anti-replay verification has not failed,
- if activation signature verification has not failed.

GA.GemActivateAtomicActivation *(GemActivate)*

Patch activation and identification are atomic operations managed by GemActivate Administrator. Operations are completely fulfilled or cancelled in case of failure.

OS.Atomicity *(OS)*
This security function performs write operations atomically on complex type or object in order to avoid incomplete update. Prior to be written, data is stored in an atomic back-up area. In case on writing interrupt, the only two possible values are: initial value if writing is not started or final value if writing is started. At next start-up, the atomic back-up area is check to finalize interrupted writing.

OS.Tests *(OS)*
This security function performs self-tests periodically to demonstrate the correct operation of security mechanisms of the IC, provides authorized users with the capability to verify the integrity of Keys, Applets, user PIN and user Keys.

OS.MemoryManagement *(OS)*
This security function allocates memory areas and performs access control on them to avoid unauthorized access. It manages circular writing to avoid instable memory state. It enforces memory recovery in case of error detection. It offers (when required) confidentiality services for data storage: Ciphering / Deciphering of Data in RAM or in FLASH, Scrambling / Unscrambling of Data in RAM or in FLASH.

OS.PatchRegistry *(OS)*
This security function manages modification of item registration in order to update the initial platform reference with patch reference to obtain the final platform reference.

OS.PatchAtomicOperation *(OS)*
This security function manages operations for patch loading, identification and activation in order to be in one of the following secure states: Patch is not loaded; Patch is loaded; Patch is loaded, identified and activated; Patch is unactivated and no more identified; Patch is deleted.

SF.REL *(Global Privacy Framework)*
This security function provides the protection of data on the TOE in the context of PACE/EAC2: Provides physical protection of the TOE and preservation of TOE secure state
Addresses the inherent leakage to TOE cryptographic operation
Provides the TOE test mechanisms
Provides protection against misuse of TOE test features

SF.AC *(Global Privacy Framework)*
This security function provides the access control of the TOE in the context of PACE/EAC2:
Provides TOE access control to specific data
Provides no access to specific data
Provides the role management
Provides management functions linked to the states of the TOE

SF.SYM_AUTH *(Global Privacy Framework)*
This security function provides the symmetric authentication functions to the TOE in the context of PACE/EAC2:
Encompasses the PACE/EAC2 identification and authentication
Manages error in SM establishment
Role authentication

SF.SM *(Global Privacy Framework)*
This security function provides the secure messaging of the TOE in the context of PACE/EAC2:
Provides the establishment of SM
Provides the secure transfer of data through SM
Performs the erasure of session keys and sensitive data
Performs high level cryptographic operations (key generation, digital signature generation, encryption/decryption in SM, random number generation, data hashing)

8.2 S3NSEN4 REV1 INTEGRATED CIRCUIT

The Security Functions (SF) introduced in this section are originate from the IC ST [ST-IC]. They realize the SFRs of the IC but are not explicitly mapped to JCS SFRs of [ST-JCS] as the IC is used in composition.

IC.LimitedFaultTolerance
The TSF provides limited fault tolerance, by managing a certain number of faults or errors that may happen, related to memory contents, CPU, random number generation and cryptographic operations, thus preventing risk of malfunction. It is related to FRU_FLT.2 from [ST-IC].

IC.SecureState
The TSF provides preservation of secure state by detecting and managing security violations, resulting in an immediate reset. It is related to FPT_FLS.1 from [ST-IC].

IC.LIM.Capability
The TSF ensures that the Secure Flash Loader and the final test capabilities are unavailable in USER configuration. It is related to FMT_LIM.1, FMT_LIM.1/Loader from [ST-IC].

IC.ModeControl
The TSF ensures that only defined modes are available: TEST, ADMIN, USER configuration. The TSF ensures the switching and the control of TOE configuration. The TSF reduces the available features depending on the TOE configuration. It is related to FMT_LIM.2, FMT_LIM.2/Loader from [ST-IC].

IC.AuditStorage
The TSF provides command to store data for audit purpose using commands only available to authorized process. It is related to FAU_SAS.1 from [ST-IC].

IC.ResistanceToPhysicalAttack
The TSF ensures resistance to physical tampering using features against probing and an active shield detecting integrity violation. It is related to FPT_PHP.3 from [ST-IC].

IC.InternalDataTransferProtection
The TSF prevents disclosure of internal and user data thanks to memory scrambling and encryption, bus encryption… It is related to FDP_ITT.1, FPT_ITT.1, FDP_IFC.1, and FDP_SDC.1 from [ST-IC].

IC.RandomNumberGeneration
The TSF produces AIS31-qualified random numbers that can be directly used in embedded software. It is related to FCS_RNG.1 from [ST-IC].

IC.CryptoOperations
The TSF provides engine to perform TDES encryption and decryption conformant to FIPS PUB 46-3. The TSF provides engine to perform AES encryption and decryption conformant to FIPS PUB 197. It is related to FCS_COP.1 [TDES] and FCS_COP.1 [AES] from [ST-IC].

IC.MemoryProtection
The TSF enforces a default memory protection policy when none other is programmed by the embedded software. It is related to FMT_MSA.3, FDP_SDI.2, FDP.ACC.1 [Memories] from [ST-IC].

IC.MPU
The TSF provides a dynamic Memory protection unit (MPU) that can be configured by the ES. It is related to FMT_MSA.1 [Memories], FMT_SMF.1 [Memories], and FDP.ACF.1 [Memories] from [ST-IC].

IC.LoadingAccessControl
The TSF provides an access control to loading. The Standard Loader instructions and/or Advanced Loader instructions can be executed only if valid passwords have been presented. It is related to FDP_ACC.1/Loader, FDP_ACF.1/Loader, FTP_ITC.1, FDP_UCT.1, FDP_UIT.1, and FIA_API.1 from [ST-IC].

8.3 TOE SUMMARY SPECIFICATION RATIONALE

Security Functional Requirement	Coverage by TSS Security Function(s)
FDP_UIT.1/CCM	**GP.CardContentManagement**, **GP.SecurityDomain** and **GP.SecureChannel** manage CCM flow control
FDP_ROL.1/CCM	**GP.CardContentManagement** and **GP.SecurityDomain** contribute to ensure that card content management operations may be properly aborted.
FDP_ITC.2/CCM	**GP.CardContentManagement** manages CCM Flow control
FPT_FLS.1/CCM	**GP.CardContentManagement** and **GP.SecurityDomain** manage failures when authentication fails and CCM fails.
FCS_COP.1/DAP	**GP.CardContentManagement** manages DAP verification
FDP_ACC.1/SD	**GP.SecurityDomain** manages access to Load File based on Delegation Token and DAP Block verification. **GP.GPRegistry** manages access and privileges to application code and instance using GP privileges.
FDP_ACF.1/SD	**GP.SecurityDomain** manages access to Load File based on Delegation Token and DAP Block verification. **GP.GPRegistry** manages access and privileges to application code and instance using GP privileges.
FMT_MSA.1/SD	**GP.SecurityDomain** allows modifying security attributes. **GP.GPRegistry** provides access to security attributes stored in the GlobalPlatform Registry.
FMT_MSA.3/SD	This SFR is covered by **GP.SecurityDomain** allowing definition of default values of security attributes. **GP.GPRegistry** provides access to security attributes stored in the GlobalPlatform Registry.
FMT_SMF.1/SD	**GP.SecurityDomain** and **GP.GPRegistry** allow modifying the behavior of security functions.
FMT_SMR.1/SD	**GP.SecurityDomain**, **GP.ISD**, **GP.SSD, GP.VASD** and **GP.CASD** manage the roles
FTP_ITC.1/SC	**GP.SecureChannel** manages secure channel and associated operations
FCO_NRO.2/SC	**GP.SecureChannel** manages secure channel
FDP_IFC.2/SC	**GP.SecureChannel** manages information flow control
FDP_IFF.1/SC	**GP.SecureChannel** manages information flow control
FMT_MSA.1/SC	**GP.SecureChannel** allows modifying security attributes
FMT_MSA.3/SC	**GP.SecureChannel** allows setting default values of security attributes
FMT_SMF.1/SC	**GP.SecurityDomain** allows performing the management functions specified in GlobalPlatform specifications using **GP.SecureChannel** to provide a secure communication channel for the transmission of the management functions.
FIA_UID.1/SC	**GP.SecureChannel** manages mutual authentication and allowed operation prior identification
FIA_UAU.1/SC	**GP.SecureChannel** manages mutual authentication and allowed operation prior identification
FIA_UAU.4/SC	**GP.SecureChannel** manages mutual authentication with anti-replay mechanism.
FDP_ACC.2/FIREWALL	This SFR is fully covered by **JCS.Firewall**
FDP_ACF.1/FIREWALL	This SFR is fully covered by **JCS.Firewall**
FDP_IFC.1/JCVM	This SFR is fully covered by **JCS.Firewall** and **JCS.APDUBuffer** controlling unauthorized access or invalid storage of reference
FDP_IFF.1/JCVM	This SFR is fully covered by **JCS.Firewall**

Security Functional Requirement	Coverage by TSS Security Function(s)
FDP_RIP.1/OBJECTS	This SFR is fully covered by **JCS.OutOfLifeDataUndisclosure** (to avoid access to data prior erase) and **JCS.EraseResidualData** (to erase data)
FMT_MSA.1/JCRE	This SFR is fully covered by **JCS.RunTimeExecution** covering context switch and application selection
FMT_MSA.1/JCVM	This SFR is fully covered by **JCS.ByteCodeExecution** requiring context switch for specific code execution and **JCS.RunTimeExecution** covering context switch and modification of the Currently Active Context according to given rules
FMT_MSA.2/FIREWALL_JCVM	This SFR is fully covered by **JCS.RunTimeExecution** covering object sharing
FMT_MSA.3/FIREWALL	This SFR is fully covered by **JCS.RunTimeExecution** covering object sharing
FMT_MSA.3/JCVM	This SFR is fully covered by **JCS.RunTimeExecution** covering object sharing
FMT_SMF.1	This SFR is fully covered by **JCS.RunTimeExecution** covering context management and instance registration
FMT_SMR.1	This SFR is fully covered by **JCS.RunTimeExecution** covering JCVM and JCRE roles
FCS_CKM.1/TDES	This SFR is fully covered by **JCS.KeyManagement** covering key generation
FCS_CKM.1/AES	This SFR is fully covered by **JCS.KeyManagement** covering key generation
FCS_CKM.1/RSA	This SFR is fully covered by **JCS.KeyManagement** covering key generation
FCS_CKM.1/ECDSA	This SFR is fully covered by **JCS.KeyManagement** covering key generation
FCS_CKM.1/HMAC	This SFR is fully covered by **JCS.KeyManagement** covering key generation
FCS_CKM.1/ECPF	This SFR is fully covered by **JCS.KeyManagement** covering key generation
FCS_CKM.1/ECDH	This SFR is fully covered by **JCS.KeyManagement** covering key generation
FCS_CKM.1/DHGen	This SFR is fully covered by **JCS.KeyManagement** covering key generation
FCS_CKM.4	This SFR is fully covered by **JCS.KeyManagement** covering key deletion
FCS_COP.1/TDES_CIPHER	This SFR is fully covered by **JCS.Crypto** covering cryptographic operation
FCS_COP.1/TDES_MAC	This SFR is fully covered by **JCS.Crypto** covering cryptographic operation
FCS_COP.1/AES_CIPHER	This SFR is fully covered by **JCS.Crypto** covering cryptographic operation
FCS_COP.1/AES_MAC	This SFR is fully covered by **JCS.Crypto** covering cryptographic operation
FCS_COP.1/RSA_SIGN	This SFR is fully covered by **JCS.Crypto** covering cryptographic operation
FCS_COP.1/RSA_CIPHER	This SFR is fully covered by **JCS.Crypto** covering cryptographic operation
FCS_COP.1/ECDSA_SIGN	This SFR is fully covered by **JCS.Crypto** covering cryptographic operation
FCS_COP.1/ECDH	This SFR is fully covered by **JCS.Crypto** covering cryptographic operation
FCS_COP.1/Hash	This SFR is fully covered by **JCS.Crypto** covering cryptographic operation
FCS_COP.1/HMAC	This SFR is fully covered by **JCS.Crypto** covering cryptographic operation
FCS_COP.1/ECDSA_KEY_GEN	This SFR is fully covered by **JCS.Crypto** covering cryptographic operation
FCS_COP.1/DH_KEY_GEN	This SFR is fully covered by **JCS.Crypto** covering cryptographic operation
FDP_RIP.1/ABORT	This SFR is fully covered by **JCS.EraseResidualData** covering data erasure
FDP_RIP.1/APDU	This SFR is fully covered by **JCS.EraseResidualData** covering data erasure
FDP_RIP.1/GlobalArray	This SFR is fully covered by **JCS.OutOfLifeDataUndisclosure** and **JCS.EraseResidualData** covering data erasure
FDP_RIP.1/bArray	This SFR is fully covered by **JCS.OutOfLifeDataUndisclosure** and **JCS.EraseResidualData** covering data erasure

Security Functional Requirement	Coverage by TSS Security Function(s)
FDP_RIP.1/KEYS	This SFR is fully covered by **JCS.EraseResidualData** covering data erasure
FDP_RIP.1/TRANSIENT	This SFR is covered by **JCS.OutOfLifeDataUndisclosure** managing the access control to transient object to be erased prior the erasure of the content in memory
FDP_ROL.1/FIREWALL	This SFR is fully covered by **JCS.RunTimeExecution** covering transaction rollback during specific operations
FAU_ARP.1	This SFR is fully covered by **JCS.RunTimeExecution**, **JCS.Exception**, **JCS.Firewall**, and **OS.MemoryManagement** covering exception handling with different specific operations
FDP_SDI.2/DATA	This SFR is fully covered by **JCS.OwnerPIN**, **JCS.KeyManagement**, **OS.Atomicity** and **OS.MemoryManagement** covering integrity handling with specific operations
FPR_UNO.1	This SFR is fully covered by **JCS.OwnerPIN, JCS.KeyManagement** and **OS.MemoryManagement** covering data handling with specific operations avoiding observation
FPT_FLS.1/JCS	This SFR is fully covered by **JCS.Exception**, **JCS.ByteCodeExecution**, **JCS.RunTimeExecution**, and **OS.Atomicity** preserving a secure state when unexpected events occur during specific operations
FPT_TDC.1	This SFR is fully covered by **JCS.Package** and **OS.MemoryManagement** assuming export check, CAP file translation and link specific operations
FIA_ATD.1/AID	This SFR is fully covered by **JCS.RunTimeExecution** and **GP.GPRegistry** controlling applet registration and uninstallation
FIA_UID.2/AID	This SFR is fully covered by **GP.GPRegistry** and **JCS.RunTimeExecution** managing user identity (package AID) during applet selection and identify associated context provided
FIA_USB.1/AID	This SFR is fully covered by **GP.GPRegistry** and **JCS.RunTimeExecution** managing registration of each applet and associated package during its installation with its AID
FMT_MTD.1/JCRE	This SFR is fully covered by **JCS.RunTimeExecution** offering services for applet registration and uninstallation managing associated access rights
FMT_MTD.3/JCRE	This SFR is fully covered by **JCS.RunTimeExecution** managing presence and legacy of AID with ISO rules
FDP_ITC.2/Installer	This SFR is fully covered by **JCS.Package** checking the binary compatibility of dependant packages using their version numbers and AIDs prior to installation operations
FMT_SMR.1/Installer	This SFR is fully covered by **JCS.RunTimeExecution**, **GP.SecurityDomain** covering the RTE, ISD and SSD roles
FPT_FLS.1/Installer	This SFR is fully covered by **JCS.Package**, **JCS.RunTimeExecution** and **GP.CardContentManagement** covering the applet instance registration operations and associated error handling
FPT_RCV.3/Installer	This SFR is fully covered by **JCS.RunTimeExecution**, **OS.MemoryManagement**, **GP.GPRegistry** and **GP.CardContentManagement** covering the applet instance erasure when applet instance registration operation fails
FDP_ACC.2/ADEL	This SFR is fully covered by **GP.CardContentManagement**, **GP.GPRegistry** and **JCS.RunTimeExecution** checking rules for applet instance uninstallation and deletion dependency rules
FDP_ACF.1/ADEL	This SFR is fully covered by **GP.CardContentManagement**, **GP.GPRegistry** and **JCS.RunTimeExecution** checking rules for applet instance uninstallation and deletion dependency rules

Security Functional Requirement	Coverage by TSS Security Function(s)
FDP_RIP.1/ADEL	This SFR is fully covered by **GP.CardContentManagement** and **JCS.OutOfLifeDataUndisclosure** by checking operations to avoid access to freed resources prior to its reuse
FMT_MSA.1/ADEL	This SFR is fully covered by **GP.GPRegistry**, **GP.CardContentManagement** and **JCS.RunTimeExecution** responsible of checking rules concerning applet attributes, implicit and explicit selection rules prior to authorize deletion operation
FMT_MSA.3/ADEL	This SFR is fully covered by **JCS.RunTimeExecution** and **GP.CardContentManagement** dealing with Security Attributes initialization, providing secure, restrictive default values for the security attributes of subject and objects involved in applet deletion
FMT_SMF.1/ADEL	This SFR is fully covered by **GP.CardContentManagement**, **GP.SecurityDomain** and **JCS.RunTimeExecution** supplying the following management functions: Modify the ActiveApplets security attribute
FMT_SMR.1/ADEL	This SFR is covered by **GP.SecurityDomain** maintaining the roles: (ISD & SDD) responsible of applet deletion. This SFR is also covered by **JCS.RunTimeExecution** maintaining the role (RTE) for applet uninstallation
FPT_FLS.1/ADEL	This SFR is covered by **GP.GPRegistry**, **JCS.RunTimeExecution** and **OS.Atomicity** preserving a secure state when unexpected events occur during package or instance deletion, managing the transaction part of the deletion operation by either rolling back, or completing it
FDP_RIP.1/ODEL	This SFR is covered by **JCS.EraseResidualData** and **OS.MemoryManagement** ensuring that the content of deleted objects is erased upon the deletion and by **JCS.OutOfLifeDataUndisclosure** making unavailable for disclosure upon further reallocation of the freed space
FPT_FLS.1/ODEL	This SFR is covered by **JCS.RunTimeExecution** and **OS.MemoryManagement** performing memory management to release no more used memory on unreferenced objects and preserves a secure state when unexpected events occur during object deletion
FCO_NRO.2/CM	This SFR is covered by **GP.SCP** managing the secure channel protocol where several checks are performed prior EF loading: * mutual authentication between the external entity (Issuer or Application provider) and the selected security Domain, including creation of a session key, * by the verification of a (chained) MAC that the Issuer or Application provider attaches to each file block sent, * by the erase of the session key at the end of the session.
FDP_IFC.2/CM	This SFR is covered by **GP.CardContentManagement** managing flow control between operations for loading, installing, selecting and executing application instances
FDP_IFF.1/CM	This SFR is covered by **GP.CardContentManagement** managing flow control between operations for loading, installing, selecting and executing application instances
FDP_UIT.1/CM	This SFR is covered by **GP.SCP** providing a session key generation. It ensures that the whole package has been correctly received
FIA_UID.1/CM	This SFR is covered by **JCS.RunTimeExecution** and **GP.SecurityDomain** controlling accessible action prior identification and action when SD or application associated to SD are selected
FMT_MSA.1/CM	This SFR is covered by **GP.SCP** setting of command security level at initialization and checking command security level during execution
FMT_MSA.3/CM	This SFR is covered by **GP.SCP** providing setting of the default value
FMT_SMF.1/CM	This SFR is covered by **GP.SCP** by setting the security level of the Secure Channel as requested by the authenticated external entity (Issuer or application provider) and updating the current value of the ICV upon reception of a new message through the Secure Channel

Security Functional Requirement	Coverage by TSS Security Function(s)
FMT_SMR.1/CM	This SFR is covered by **GP.SecurityDomain**, **GP.ISD**, **GP.SSD, GP.VASD** and **GP.CASD** managing the roles: issuer, application provider, verification authority and controlling authority
FTP_ITC.1/CM	This SFR is covered by **GP.SCP** for applet loading
FPT_TST.1/SCP	This SFR is covered by **OS.Tests**
FPT_RCV.3/SCP	This SFR is covered by **OS.Atomicity**
FPT_PHP.3/SCP	This SFR is covered by **IC.ResistanceToPhysicalAttack**
FPT_RCV.4/SCP	This SFR is covered by **OS.MemoryManagement**
FCS_RNG.1	This SFR is covered by **JCS.RNG** providing a dedicated API to applet. **JCS.RNG** uses **IC.RandomNumberGeneration** to supply the service
FPT_FLS.1/SecureAPI	This SFR is fully covered by **SA.SecureOperation**, **SA.FlowControl** and **IC.LimitedFaultTolerance**
FPT_ITT.1/SecureAPI	This SFR is fully covered by **SA.SecureOperation** and **IC.InternalDataTransferProtection**
FPR_UNO.1/SecureAPI	This SFR is fully covered by **SA.RandomDelay**, **SA.SecureOperation** and **IC.RandomNumberGeneration**
FMT_SMR.1/GemActivate	This SFR is fully covered by **GA.GemActivateActivation** maintaining GemActivate Administrator role
FMT_SMF.1/GemActivate	This SFR is fully covered by **GA.GemActivateActivation**
FMT_MOF.1/GemActivate	This SFR is fully covered by **GA.OptionalServiceActivation**
FMT_MSA.1/GemActivate	This SFR is fully covered by **GA.OptionalServiceActivation**
FMT_MTD.1/GemActivate	This SFR is fully covered by **GA.ServiceAudit**
FDP_ACC.1/GemActivate	This SFR is fully covered by **GA.GemActivateActivation**
FDP_ACF.1/ GemActivate	This SFR is fully covered by **GA.GemActivateActivation**
FMT_MSA.3/GemActivate	This SFR is fully covered by **GA.GemActivateActivation**
FMT_TRP.1/OS-UPDATE	This SFR is fully covered by **GA.PatchManagement**
FIA_ATD.1/OS-UPDATE	This SFR is fully covered by JCS.RunTimeExecution, GP.OS-UPDATE_CODE_ID, GP.GPRegistry, GA.PatchManagement, GA.GemActivateAtomicActivation, OS.PatchRegistry and OS.PatchAtomicOperation controlling patch registration and uninstallation.
FCS_CKM.1/DH_PACE	This SFR is fulfilled by SF.SM “Secure Messaging” which enforces PACE/EAC2 SM cryptographic mechanisms
FCS_CKM.4/PACE	This SFR is fulfilled by SF.SM.
FCS_COP.1/PACE_ENC	This SFR is fulfilled by SF.SM “Secure Messaging” which enforces PACE/EAC2 SM cryptographic mechanisms.
FCS_COP.1/PACE_MAC	This SFR is fulfilled by SF.SM “Secure Messaging” which enforces PACE/EAC2 SM cryptographic mechanisms
FCS_COP.1/PACE_CAM	This SFR is fulfilled by SF.SM “Secure Messaging” which enforces PACE/EAC2 SM cryptographic mechanisms
FCS_RNG.1/PACE	This SFR is fulfilled by SF.SM managing RND generation
FIA_AFL.1/PACE	This SFR is fulfilled by SF.SYM_AUTH “Symmetric authentication” which manages symmetric authentication functions and error management

Security Functional Requirement	Coverage by TSS Security Function(s)
FIA_UID.1/PACE	This SFR is fulfilled by SF.SYM_AUTH “Symmetric authentication” which manages symmetric authentication functions and error management
FIA_UAU.1/PACE	This SFR is fulfilled SF.SYM_AUTH “Symmetric authentication” which manages symmetric authentication functions and error management
FIA_UAU.4/PACE	This SFR is fulfilled SF.SYM_AUTH “Symmetric authentication” which manages symmetric authentication functions and error management
FIA_UAU.5/PACE	This SFR is fulfilled by SF.SYM_AUTH “Symmetric authentication” which manages symmetric authentication functions and error management
FIA_UAU.6/PACE	This SFR is fulfilled by SF.SYM_AUTH “Symmetric authentication” which manages symmetric authentication functions and error management
FDP_RIP.1/PACE	This SFR is fulfilled by SF.SM “Secure Messaging” which enforce the erasure of sensitive data transferred in secure channel
FTP_ITC.1/PACE	This SFR is fulfilled by SF.SM “Secure Messaging” which ensures the establishment of the secure messaging.
FMT_SMF.1/PACE	This SFR is fulfilled by SF.AC “Access Control” which ensures the management functions in the different life cycle status.
FMT_SMR.1/PACE	This SFR is fulfilled by SF.AC “Access Control” which maintains the different roles according to the life cycle status. It is also fulfilled by SF.SYM_AUTH “Symmetric authentication” which authenticate roles.
FMT_LIM.1/PERSO	This SFR is fulfilled by SF.AC “Access Control” which limit the capabilities and availability of the TSF after TOE delivery.
FMT_LIM.2/PERSO	This SFR is fulfilled by SF.AC “Access Control” which limit the capabilities and availability of the TSF after TOE delivery.
FMT_MTD.1/INI_ENA	This SFR is fulfilled by SF.AC “Access Control” which manages the access control.
FMT_MTD.1/INI_DIS	This SFR is fulfilled by SF.AC “Access Control” which manages the access control.
FMT_MTD.1/KEY_READ	This SFR is fulfilled by SF.AC “Access Control” which manages the access control.
FPT_EMS.1	This SFR is fulfilled by SF.REL ”Reliability” which implements measures to limit information contained in electromagnetic and current emissions.
FPT_FLS.1	This SFR is fulfilled by SF.REL “Reliability” which preserves secure states.
FPT_TST.1	This SFR is fulfilled by SF.REL “Reliability” which implements tests to protect the TOE.
FPT_PHP.3	This SFR is fulfilled by the TOE security function SF.REL “Reliability” which protects the TOE against physical attacks.
FCS_COP.1/SHA	This SFR is fulfilled by SF.SM which provides Data Hashing
FCS_COP.1/SIG_VER	This SFR is fulfilled by SF.SM which provides signature verification
FIA_API.1/CA	This SFR is fulfilled by SF.SM “Secure Messaging” which enforces EAC2 SM cryptographic mechanisms
FIA_UID.1/EAC2_Terminal	This SFR is fulfilled by SF.SYM_AUTH and SF.AC that provide user identification and user authentication prior to enabling access to authorized functions.

Security Functional Requirement	Coverage by TSS Security Function(s)
FIA_UAU.1/EAC2_Terminal	This SFR is fulfilled by SF.SYM_AUTH “Symmetric authentication” and SF.AC which provide user identification and user authentication prior to enabling access to authorized functions. It is also met by SF.SM “Secure Messaging” which ensures the establishment of the secure messaging.
FIA_UAU.6/CA	This SFR is covered by SF.SYM_AUTH and SF.AC that provide user identification and user authentication prior to enabling access to authorized functions, and by SF.SM “Secure Messaging” which ensures the establishment of the secure messaging.
FTP_ITC.1/CA2	This SFR is covered by SF.SYM_AUTH and SF.AC that enforce the access right policy for data exchange between the TOE and an EAC2 terminal and by SF.SM which ensures the establishment of the secure channel between the TOE and an EAC2 terminal to protect the exchanged data from modification and disclosure.
FMT_MTD.1/Initialize_PIN	This SFR is fulfilled by SF.AC “Access Control” which manages the access control and ensures that only authenticated personalization agent can write PIN, PUK, MRZ and CAN.

END OF DOCUMENT