Public Security Target

DAKOTA IOT V1.0

on SLI37-SLM37

Edition: 2

DOCUMENT EVOLUTION

Issue	Date	Author	Purpose
1	18/10/2024	IDEMIA	Creation from FQR 110 A2D5 Ed1
2	20/11/2024	IDEMIA	Issue for certification with IC certificate update

Table of contents

Table of figures

Table of tables

1 References and Abbreviations

1.1 References

Ref	Title
[1]	Java Card™ System - Open Configuration Protection Profile, version 3.1, April 2020, BSI-CC-PP-0099-V2-2020.
[2]	Security IC Platform Protection Profile with Augmentation Packages version 1.0, February 2014, BSI-CC-PP-0084-2014.
[3]	GSMA SGP.02 - Remote Provisioning Architecture for Embedded UICC Technical Specification v4.1
[4]	(U)SIM Java Card Platform Protection Profile Basic and SCWS Configurations, version 2.0.2, July 2010, ANSSI-CC-PP-2010/05.
[5]	GlobalPlatform Card Composition Model Security Guidelines for Basic Applications, version 2.0, December 2014 – ref. GPC_GUI_050.
[6]	ETSI TS 102 221 - Smart Cards; UICC-Terminal interface; Physical and logical characteristics (Release 16).
[7]	Joint Interpretation Library – The application of CC to integrated circuits, version 3.0, February 2009.
[8]	Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model, version 3.1, Revision 5, April 2017.
[9]	Common Criteria for Information Technology Security Evaluation, Part 2: Security functional components, version 3.1, Revision 5, April 2017.
[10]	Common Criteria for Information Technology Security Evaluation, Part 3: Security assurance components, version 3.1, Revision 5, April 2017.
[11]	GlobalPlatform Card Specification v2.3 including • Card Confidential Card Content Management Card Specification v2.3 - Amendment A v1.1; • Card Remote Application Management over HTTP Card Specification v2.3 – Amendment B v1.2; • Card Technology Secure Channel Protocol '03' Card Specification v2.2 – Amendment D V1.1.1; • Security Upgrade for Card Content Management, Card Specification v2.2 - Amendment E" V1.0 - Document Reference: GPC_SPE_042 • Secure Channel Protocol '11'(SCP11c) – Amendment F v1.2.1
[12]	SP80 - ETSI TS 102 225 - Secured packet structure for UICC based applications, version 12.0.0, release 12. ETSI TS 102 226 - Remote APDU structure for UICC based applications, version 12.0.0, release 12.
[13]	SP81- GlobalPlatform Card Specification Amendment B – Remote Application Management over HTTP, version 1.1.3, May 2015.

Ref	Title
[14]	Joint Interpretation Library – Composite Product Evaluation for Smart Cards and similar devices, Version 1.5.1, May 2018.
[15]	3GPP TS 43.019 - Subscriber Identity Module Application Programming; Interface (SIM API) for Java Card, version 6.0.0, release 6, December 2004.
[16]	ETSI TS 102 241 - UICC Application Programming Interface (UICC API) for Java Card, version 17.5.1, release 17.
[17]	3GPP TS 31.130 - (U)SIM API for Java™ Card, version 9.4.0, release 9, April 2012.
[18]	3GPP TS 31.133 - ISIM API for Java Card™, version 9.2.0 - release 9, May 2011.
[19]	W. Killmann, W. Schindler, „A proposal for: Functionality classes for random number generators‟, version 2.0, September, 2011.
[20]	3GPP TS 35.205, 3GPP TS 35.206, 3GPP TS 35.207, 3GPP TS 35.208, 3GPP TR 35.909 (Release 11): "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Specification of the MILENAGE Algorithm Set: An example algorithm set for the 3GPP authentication and key generation functions f1, f1*, f2, f3, f4, f5 and f5*; • Document 1: General; • Document 2: Algorithm Specification; • Document 3: Implementers Test Data; • Document 4: Design Conformance Test Data; • Document 5: Summary and results of design and evaluation.
[21]	Tuak- 3GPP TS 35.231, 3GPP TS 35.232, 3GPP TS 35.233, version 12.1.0 , release 12, December 2014. • Document 1: Algorithm specification; • Document 2: Implementers' test data; • Document 3: Design conformance test data.
[22]	3GPP TS 33.102, 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Security architecture, version 16.2.0. 3GPP TS 33.401, 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3GPP System Architecture Evolution (SAE); Security architecture, version 12.16.0, release 12, December 2015.
[23]	Remote SIM Provisioning (RSP) Architecture, version 2.3, GMSA Association, June 2021 - SGP.21
[24]	Remote SIM Provisioning (RSP) Technical Specification, version 2.3, GSMA Association, June 2021 - SGP.22.
[25]	3GPP TS 23.003 version 15.3.0 - Numbering, addressing and identification (Release 15).
[26]	GSMA TS.26 – NFC Handset Requirements, version 11.0, June 2017.
[27]	Embedded UICC for Consumer Devices Protection Profile, BSI-CC-PP-0100 (GSMA SGP.25 v1.0)
[28]	Common Methodology for Information Technology Security Evaluation, version 3.1, Revision 5, April 2017.

Ref	Title
[29]	Java Card Platform, versions 3.1, Classic Edition, Runtime Environment (Java Card RE) Specification. Published by Oracle. [JAVASPEC]
[30]	Java Card Platform, versions 3.1, Classic Edition, Virtual Machine (Java Card VM) Specification. Published by Oracle.
[31]	No more used
[32]	Java Card Platform, versions 3.1, Classic Edition, Application Programming Interface (API). Published by Oracle.
[33]	The Java Virtual Machine Specification. Lindholm, Yellin ISBN 0-201-43294-3
[34]	IEEE Std 1363a-2004 Standard Specification of Public-Key Cryptography
[35]	GlobalPlatform Card Technology - Secure Channel Protocol '03', Card Specification v2.2 – Amendment D" Version 1.1.1 - Public Release July 2014 Document Reference: GPC_SPE_014
[36]	No more used
[37]	"FIPS PUB 81, DES Modes of Operation" April 17, 1995, National Institute of Standards and Technology
[38]	No more used
[39]	"GlobalPlatform Card Specification" Version 2.3 Public Release - October 2015 Document Reference: GPC_SPE_034
[40]	See [35]
[41]	IEEE Std 1363a-2004 Standard Specification of Public-Key Cryptography
[42]	No more used
[43]	Certification of « open » smart card products, Version 1.1 (for trial use), 4 February 2013.
[44]	3GPP TS 33.501 V15.4.0 Security architecture and procedures for 5G system - ETSI TS 133 501 V15.4.0.
[45]	SECG specifications SEC 1: Elliptic Curve Cryptography v2.0
[46]	SECG specifications SEC 2: Recommended Elliptic Curve Domain Parameters v2.0
[47]	3rd Generation Partnership Project; Technical Specification Group Services and System Aspects. Security architecture and procedures for 5G system (Release 16)
[48]	The NIST SP 800-90 Recommendation for Random Number Generation Using Deterministic Random Bit Generators (Revise) March 2007
[49]	ETSI TS 102 223 - Smart Cards; Card Application Toolkit (CAT) Release 12.
[50]	Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Technical realization of the Short Message Service (SMS) 3GPP TS 23.040
[51]	GSMA SGP.32 –eSIM IoT Technical Specification Version 1.0.1 04 July 2023
[52]	Appendix to be added to the COMP 128-2 specification document to produce COMP128-3. GSM-A document, Reference: SG Doc 114/01, GSM Association.

Ref	Title
[53]	eUICC Profile Package Interoperable Format Technical Specification version 2.3.1, Trusted Connectivity Alliance – TCA.

IDEMIA internal references

Ref	Title
[AGD_SR]	DaKota IoT v1.0 on SLx37 - Applet Security Recommendations, FQR 110 A2FB Ed 2
[AGD_OPE]	DaKota IoT v1.0 on SLx37 AGD_OPE, FQR 110 A300 Ed1
[AGD_PAPI]	DaKota IoT v1.0 on SLx37 Javadoc, FQR 110 A318 Ed1
[AGD_PRE]	DaKota IoT v1.0 on SLx37 AGD_PRE, FQR 110 A2F9 Ed1
[AGD_ALP]	DaKota IoT v1.0 on SLx37 Application Loading Protection Guidance, FQR 110 A2FD Ed1
[JPATCH]	DaKota IoT - JCVM Patch Loading Protection Guidance, FQR 110 A30B Ed1

1.2 Abbreviations

Abbreviation	Description
AID	Application Identifier
ASN.1	Abstract Syntax Notation One
BIP	Bearer Independent Protocol
CA	Certificate Authority
CERT.CI.ECDSA	Certificate of the CI for its Public ECDSA Key
CERT.DPauth.ECDSA	Certificate of the SM-DP+ for its Public ECDSA key used for SM-DP+ authentication
CERT.DPpb.ECDSA	Certificate of the SM-DP+ for its Public ECDSA key used for Profile Package Binding
CERT.DSauth.ECDSA	Certificate of the SM-DS for its Public ECDSA key used for SM-DS authentication
CERT.EUICC.ECDSA	Certificate of the eUICC for its Public ECDSA key
CERT.EUM.ECDSA	Certificate of the EUM for its Public ECDSA key
CERT.DP.TLS	Certificate of the SM-DP+ for securing TLS connections (version >= 1.2)
CERT.DS.TLS	Certificate of the SM-DS for securing TLS connections (version >= 1.2)
CI	Certificate Issuer
CMAC	Cipher-based MAC
CRL	Certificate Revocation List
DH	Diffie-Hellman
ECASD	eUICC Controlling Authority Security Domain
ECC	Elliptic Curve Cryptography

Abbreviation	Description
ECDSA	Elliptic Curve cryptography Digital Signature Algorithm
ECKA	Elliptic Curve cryptography Key Agreement algorithm
EID	eUICC-ID
ETSI	European Telecommunications Standards Institute
eUICC	Embedded Universal Integrated Circuit Card
EUM	eUICC Manufacturer
GP	GlobalPlatform
GSMA	GSM Association
HLR	Home Location Register
ICCID	Integrated Circuit Card ID
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
ISD	Issuer Security Domain
IPA	IoT Profile Assistant
IPAe	IoT Profile Assistant located in the eUICC
IPAd	IoT Profile Assistant located in the IoT Device
ISD-P	Issuer Security Domain Profile
ISD-R	Issuer Security Domain Root
ISK	Initialization Secret Key
JSK	JPatch Secret Key:
ISO	International Standards Organisation
ITU	International Telecommunications Union
LDS	Local Discovery Service
LPA	Local Profile Assistant
LPAd	Local Profile Assistant when LPA is in the Device
LPAe	Local Profile Assistant when LPA is in the eUICC
LTE	Long Term Evolution
LUIe	Local User Interface when LPA is in the eUICC
MAC	Message Authentication Code
MNO	Mobile Network Operator
MSK	Master Secret Key
NAA	Network Access Application
OTA	Over The Air
otPK.DP.ECKA	One-time Public Key of the SM-DP+ for ECKA
otPK.EUICC.ECKA	One-time Public Key of the eUICC for ECKA
otSK.DP.ECKA	One-time Private Key of the SM-DP+ for ECKA

Abbreviation	Description
otSK.EUICC.ECKA	One-time Private Key of the eUICC for ECKA
PE	Profile Element
PKI	Public Key Infrastructure
PK.CI.ECDSA	Public Key of the CI, part of the CERT.CI.ECDSA
PK.DPauth.ECDSA	Public Key of the SM-DP+ part of the CERT.DPauth.ECDSA
PK.DPpb.ECDSA	Public Key of the SM-DP+ part of the CERT.DPpb.ECDSA
PK.DSauth.ECDSA	Public Key of the SM-DS part of the CERT.DSauth.ECDSA
PK.EUICC.ECDSA	Public Key of the eUICC, part of the CERT.EUICC.ECDSA
PK.EUM.ECDSA	Public Key of the EUM, part of the CERT.EUM.ECDSA
POS	Point Of Sale
PPI	Profile Package Interpreter
PPE	Profile Policy Enabler
PPR	Profile Policy Rule
RAT	Rules Authorisation Table
RSP	Remote SIM Provisioning
SAS	Security Accreditation Scheme
SCP	Secure Channel Protocol
SD	Security Domain
S-ENC	Session key for message encryption/decryption
S-MAC	Session Key for message MAC generation/verification
ShS	Shared Secret
SK.CI.ECDSA	Private key of the CI for signing certificates
SK.DPauth.ECDSA	Private Key of the of SM-DP+ for creating signatures for SM-DP+ authentication
SK.DPpb.ECDSA	Private key of the SM-DP+ used to provide signatures for Profile binding
SK.DSauth.ECDSA	Private Key of the of SM-DS for creating signatures for SM-DS authentication
SK.EUICC.ECDSA	Private key of the eUICC for creating signatures
SK.EUM.ECDSA	Private key of the EUM for creating signatures
SK.DP.TLS	Private key of the SM-DP+ for securing TLS connection connections (version >= 1.2)
SK.DS.TLS	Private key of the SM-DS for securing TLS connection connections (version >= 1.2)
SM-DP+	Subscription Manager Data Preparation (Enhanced compared to the SM-DP in SGP.02 [3])
SM-DS	Subscription Manager Discovery Server
SUCI	The SUbscription Concealed Identifier

Abbreviation	Description
SUPI	The SUbscription Permanent Identifier
SVN	SGP.22 Specification Version Number (referred to as 'eSVN' in SGP.21 [23]).
TLS	Transport Layer Security (version >= 1.2)
USIM	Universal Subscriber Identity Module

1.3 Technical Terms

Term	Description
Alternative SM-DS	SM-DS used in cascade mode with a Root SM-DS to redirect Event Registration from an SM-DP+ to the Root SM-DS.
Certificate Authority	A Certificate Authority is an entity that issues digital certificates.
Certificate Issuer	An Entity that is Authorized to Issue digital certificates.
Device	User equipment used in conjunction with an eUICC to connect to a mobile network. E.g. a tablet, wearable, smartphone or handset.
DAP Block	Part of the Load File used for ensuring Load File Data Block verification.
DAP Verification	Mechanism used by a Security Domain to verify that a Load File Data Block is authentic.
Disabled (Profile)	The state of a Profile where all files and applications (e.g. NAA) present in the Profile are not selectable over the eUICC- Terminal interface.
Embedded UICC	A UICC which is not easily accessible or replaceable, is not intended to be removed or replaced in the Device, and enables the secure changing of Profiles.
Enabled (Profile)	The state of a Profile when its files and/or applications (e.g., NAA) are selectable over the UICC-Terminal interface.
eUICC Certificate	A certificate issued by the EUM for a specific eUICC. This Certificate can be verified using the EUM Certificate.
eUICC Manufacturer	Supplier of the eUICCs and resident software (e.g. firmware and operating system).
Event	A Profile download which is set by an SM-DP+ on behalf of an Operator, to be processed by a specific eUICC.
EventID	Unique identifier of an Event for a specific EID generated by the SM-DP+ / SM-DS.
Event Record	The set of information stored on the SM-DS for a specific Event, via the Event Registration procedure. This information consists of either: • the Event-ID, EID, and SM-DP+ address or the Event-ID, EID, and SM-DS address.
Event Registration	A process notifying the SM-DS on the availability of information on either a specific SM-DP+ or a specific SM-DS for a specific eUICC.
EUM Certificate	A certificate issued to a GSMA accredited EUM which can be used to verify eUICC Certificates. This Certificate can be verified using the Root Certificate.
Integrated Circuit Card ID	Unique number to identify a Profile in an eUICC. Note: the ICCID throughout this specification is used to identify the Profile.

Term	Description
International Mobile Subscriber Identity	Unique identifier owned and issued by Mobile operators to (U)SIM applications to enable Devices to attach to a network and use services as defined in 3GPP TS 23.003 [25] section 2.2.
Issuer Identifier Number	The first 8 digits of the EID.
Issuer Security Domain	A security domain on the UICC as defined by GlobalPlatform Card Specification [11].
JSK	Secret key used for patch loading.
Local Profile Management	Local Profile Management are operations that are locally initiated on the End User (ESeu) interface.
Local Profile Management Operation	Local Profile Management Operations include enable Profile, disable Profile, delete Profile, query Profile Metadata, eUICC Memory Reset, eUICC Test Memory Reset and Set Nickname.
Load File	The Load File is the data that represents the cap file to be loaded in the product.
Load File Data Block Hash	The Load File Data Block Hash provides integrity of the Load File Data Block following receipt of the complete Load File Data Block.
MatchingID	Equivalent to "Activation Code Token" as defined in SGP.21 [23]: "A part of the Activation Code information provided by the Operator/Service Provider to reference a Subscription".
Mobile Network Operator	An entity providing access capability and communication services to its End User through a mobile network infrastructure.
Mobile Network Operator Security Domain (MNO-SD)	Part of the Profile, owned by the Operator, providing the Secured Channel to the Operator's Over The Air (OTA) Platform. It is used to manage the content of a Profile once the Profile is enabled.
NFC Device	A Device compliant with GSMA TS.26 [26].
Notification	A report about a Profile download and Local Profile Management Operation processed by the eUICC.
Operational Profile	A Profile that allows connectivity to a commercial mobile network.
OTA Keys	The credentials included in the Profile, used in conjunction with OTA Platforms.
OTA Platform	An Operator platform for remote management of UICCs and the content of Enabled Operator Profiles on eUICCs.
Profile	Combination of a file structure, data and applications to be provisioned onto, or present on, an eUICC and which typically allows, when enabled, the access to a specific network A Profile can be an Operational, Provisioning or Test Profile.
Profile Component	A Profile Component is an element of the Profile, when installed in the eUICC, and MAY be one of the following: • An element of the file system like an MF, EF or DF;

Term	Description
	• An Application, including NAA and Security Domain; • Profile metadata, including Profile Policy Rules; • An MNO-SD.
Profile Management	A set of functions related to the downloading, installation and content update of a Profile in a dedicated ISD-P on the eUICC. Download and installation are protected by Profile Management Credentials shared between the SM-DP+ and the ISD-P.
Profile Management Credentials	Data required within an eUICC so that a Profile downloaded from an external entity can be decrypted and installed on the eUICC.
Profile Managemen t Operation	Local or Remote Profile Management operation: Enable Profile, Disable Profile, Delete Profile
Profile Nickname	Alternative name of the Profile set by the End User.
Profile Policy Authorisation Rule	A set of data that governs the ability of a Profile Owner to make use of a Profile Policy Rule in a Profile.
Profile Policy Rule	Defines a qualification for or enforcement of an action to be performed on a Profile when a certain condition occurs.
Profile Type	Operator specific defined type of Profile. This is equivalent to the "Profile Description ID" as described in Annex B of SGP.21 [23]
Provisioning Profile	A Profile that allows connectivity to a commercial mobile network solely to provide system services, such as the provisioning of Profiles.
Roles	Roles are representing a logical grouping of functions.
Root SM-DS	A globally identified central access point for finding Events from one or more SM-DP+(s).
Rules Authorization Table	A set of Profile Policy Authorisation Rules that, together, determines the ability of a Profile Owner to make use of a set of Profile Policy Rules in a Profile.
SCP-SGP22	Protocol for Profile Protection and eUICC Binding defined in [24] and based on SCP11 ([11] Amendment F)
Service Provider	The organization through which the End User obtains PLMN telecommunication services. This is usually the network operator or possibly a separate body.
SM-DP+ OID	Identifier of the SM-DP+ that is globally unique and is included as part of the SM-DP+ Certificate.
SM-DS OID	Identifier of the SM-DS that is globally unique and is included as part of the SM-DS Certificate.
Subscription	Describes the commercial relationship between the End User and the Service Provider.
Subscription Manager Data Preparation+ (SM-DP+)	This role prepares Profile Packages, secures them with a Profile protection key, stores Profile protection keys in a secure manner and the Protected

Term	Description
	Profile Packages in a Profile Package repository, and allocates the Protected Profile Packages to specified EIDs. The SM-DP+ binds Protected Profile Packages to the respective EID and securely downloads these Bound Profile Packages of the respective eUICC.
Subscription Manager Discovery Server (SM-DS)	This is responsible for providing addresses of one or more SM-DP+(s) to an LDS.
Test Profile	A Profile used for the purpose of testing the Device and the eUICC. A Test Profile will not include any Operator Credentials.
User Intent	Describes the direct, real time acquisition and validation of the manual End User instruction on the LUI to trigger locally a Profile download or Profile Management operation. As defined in SGP.21 [23].

2 Introduction

This document defines the Public Security Target for the remote provisioning and management of the Embedded UICC in Consumer Devices, following the modular approach from [8].

2.1 Security Target Reference

Title	DAKOTA IOT Security Target on SLI37-SLM37
ST Identification	FQR 110 A2D5
Public ST Identification	FQR 110 A3A5
ST Version	Ed 2
Public ST Version	Ed 2
CC Version	3.1 Revision 5
Assurance Level	EAL4 augmented with ALC_DVS.2, ALC_FLR.1 and AVA_VAN.5
Compliant To Protection Profile	[27]
ITSEF	Serma
Certification Body	GSMA - TrustCB

Table 1: ST Reference

2.2 TOE Reference

Product Commercial Name	DAKOTA IoT v1.0
TOE Name	DAKOTA IoT v1.0 on SLx37
TOE Software version	SAAAAR Code: 09A451
IC	Infineon SLI37 and SLM37 certificate: NSCIB-CC-2200060-02
TOE Guidance	See Reference Guidance in Table 3: Guidance references

Table 2: TOE Reference

The applet identifiers (AID) are:

- **5G SUCI:** A0000000 871005FF FFFFFF89 13400000
- **BIPLink:** A0000000 77010000 1A100000 00000002

The TOE is identified by the tag identity, which provides information on the product and allows identifying each product configuration in term of features included or not in each specific product configuration. Information and values to identified TOE are described in Reference guidance.

From the Infineon SLI37 and SLM37 certificate: NSCIB-CC-2200060-02 certificate, the TOE considers only the following references:

- IO417 - MFF2 - SLI37CCA1M0 - IC Type = '6705'
- IO419 - SimFit - SLM37ECA1M0 - IC type = '6609'

Audience	Ref	Form factor of delivery
Guidance for developer of sensitive applications	**[AGD_SR]**	Electronic version
Guidance for application developer	**[AGD_OPE]** **[AGD_PAPI]**	
Guidance for using JPATCH and patch loading	**[JPATCH]**	
Guidance for preparative procedures	**[AGD_PRE]**	
Issuer of the platform that aims to load applications	**[AGD_ALP]**	

Table 3: Guidance references

AGD presentation:

[AGD_SR]

If the applet needs to have a security certification, the applet must follow recommendations listed in the document.

If the applet is a basic application, and does not need security certification with the platform, the certificate of the Platform is still valid if the applet go through the verifier before the loading of this applet (the security function of the platform are still available).

This guide is provided to the developer and evaluator of a sensitive application to be certified.

[AGD_OPE]

This document describes the TOE card usage. It describes how to use the card from an APDU commands point of view and gets onto topics such as common platform APDU commands, secure channels and security domains.

This document also describes the available Java Card and proprietary APIs for applet developers.

This guide is provided to the users, personalizer and developer of Java Card applications to be certified or not. It does not mandate any requirement for the developer; it constitutes a help.

[AGD_PRE]

This document is aimed to describe the AGD PRE (Preparative procedures) for Dakota IOT products. It describes the preparative steps that should be followed to correctly initialize the platform. The TOE is finalized once it is prepared.

The TOE is finalized once it is prepared.

[AGD_ALP]

This document describes the loading procedure, in compliance with ANSSI Application Note 10 and the Java Card Open Platform protection profile.

[AGD_PAPI]

This document summarizes the ID proprietary API (packages, classes, methods and fields) available on the Java Card Identity Platforms.

This guide is provided to the Developer of Java Card applications to be certified or not. It does not mandate any requirement for the developer; it constitutes a help.

[JPATCH]

Guidance for using JPATCH and patch loading. This guidance describes the patch traceability.

This guidance is provided to the entity that has the card content management rights.

The Guidance is aimed to be used by IDEMIA R&D. The patch has to be developed only by IDEMIA R&D. Any patch must be evaluated:

- by maintenance process if the patch does not impact the security

- or by reassessment if the intended patch impacts the security of any of evaluated security function of the present scope.

2.3 TOE overview

The TOE consists of the embedded UICC (eUICC) IC and software based on Javacard 3.1 that implements the GSMA Remote SIM Provisioning (RSP) Architecture for Consumer Devices ([23] and [24]).

The TOE includes:

- The Application Layer: privileged applications, such as Security Domains, providing the remote provisioning and administration functionality (the notion of Security Domain follows the definition given by [11]):
  - An ISD-R, providing life-cycle management of profiles.
  - An ECASD providing secure storage of credentials and security functions for key establishment and eUICC authentication.
  - ISD-P security domains, each one hosting a unique profile.
- The Platform Layer: a set of functions providing support to the Application Layer:
  - A Telecom Framework providing network authentication algorithms.
  - A Profile Package Interpreter translating Profile Package data into an installed Profile.
  - And a Profile Policy Enabler which comprises Profile Policy verification and enforcement functions.
  - A runtime environment based on JavaCard.
- The secure IC and its embedded software.

The TOE relies on a Local Profile Assistant (LPA) component. It is implemented at the application level as LPAe (covered by the LPA PP-Module), or its implemented as a non-TOE on-device unit called LPAd (dependent of the used device).

In the personalization phase, the personaliser defines the way to use: LPAd or LPAe.

LPAe is part of the product **and is not in the scope of this evaluation**.

2.3.1 TOE type and TOE major security features

The TOE type is software embedded on an IC.

The eUICC is embedded in a consumer device. Whether the eUICC has a form factor enabling replacement is not considered here: the eUICC could be removable once it is rolled out. The eUICC is connected to a given mobile network, by the means of its currently enabled MNO Profile.

The underlying IC is also a part of the TOE and is covered by a separate IC certification.

The product embeds an IOT overlay out of scope of the TOE. For this, the IoT Profile Assistant named IPA Agent is embedded in the product, not instantiated and out of scope of the TOE.

Figure 1 below shows the scope of the TOE:

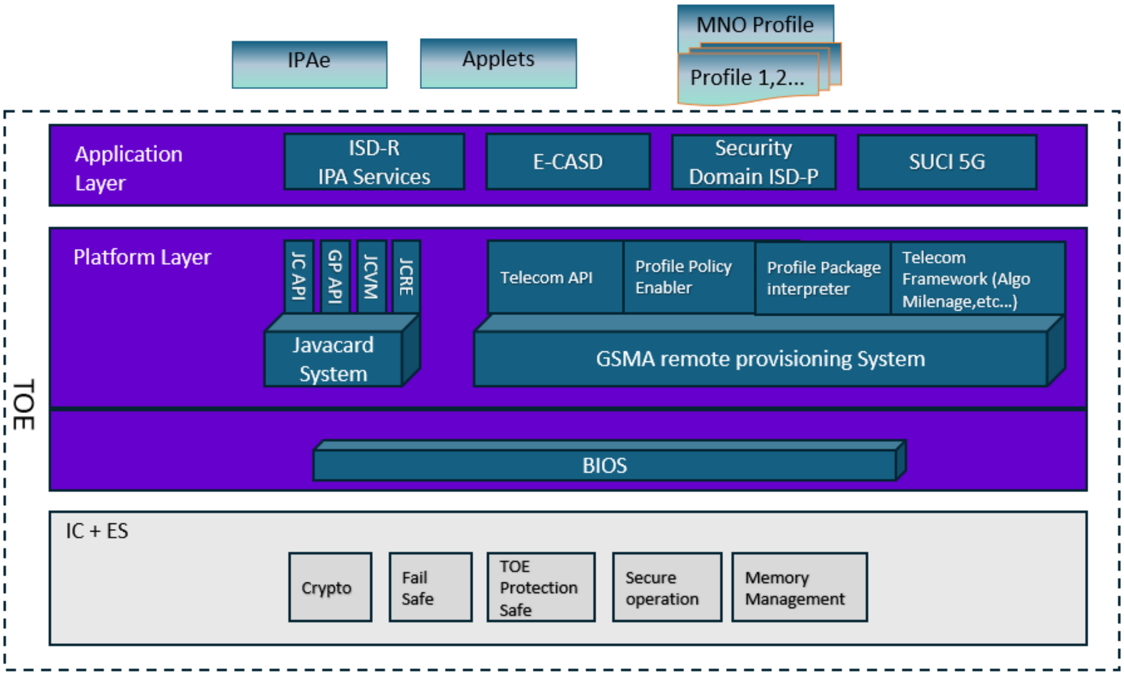

Figure 1: Scope of the TOE

The functional level of the OS is based on a Java™ based multi-application open platform, compliant with Java Card 3.1 Classic Edition and Global Platform 2.3 specifications.

This platform is able to receive and manage different types of applications; i.e. Basic and Sensitive ones.

All the platform code including GP Java application called card manager are loaded in the FLASH memory.

The TOE allows the loading of optional code, Java Card application and native code:

1. Applications can be loaded on the flash memory, at pre-personalisation, personalisation or use phase.
2. Optional code can be loaded to upgrade the TOE at any time of product life cycle this function is named JPatch.

However, the card issuer (installer) can forbid each of these operations before or after the issuance of the IC.

The mechanism for the different loading is part of the present ST and is also part of the TOE evaluation.

2.3.2 TOE usage

The eUICC will contain several MNO Profiles, each of them being associated with a given International Mobile Subscriber Identity (IMSI).

The primary function of the Profile is to authenticate the validity of a Device when accessing the network. The Profile is MNO's property, and stores MNO specific information.

An eUICC with an enabled operational Profile provides the same functionality as a SIM or USIM card.

2.3.3 Non-TOE HW/SW/FW Available to the TOE

2.3.3.1 Description of Non-TOE HW/FW/SW and systems

The Non-TOE HW/FW/SW and systems is the one described in PP [27] with the following specificities:

The eUICC is included in a device with IOT overlay out of scope of the TOE with respect of corresponding specifications SGP.32 [51].

MNO-SD and applications

The application IPAe as defined in SGP.32 is embedded in the product but out of scope of the TOE, The IPAe is described in the PP [27] section 1.2.4.2.

NOTE: RMI functions are not implemented by the TOE.

2.4 PRODUCT ARCHITECTURE

2.4.1 Logical scope of the TOE

As show in Figure 1 with dotted line, the TOE includes the BIOS, the Virtual Machine, the APIs, the Global Platform application (with the CM), the Javacard Runtime and Firewall mechanism, the Framework application and the IC component. The TOE integrates also patch mechanism called Jpatch, implemented in the VM block. Th applications embedded are the Telecom applications dedicated to consumer device.

2.4.1.1 Application Layer

Profiles management

The goal of the Application layer is to implement the eUICC functionalities described in [23] and [24], which rely on the notion of a Profile. A Profile is the combination of a file structure, data and applications to be provisioned onto, or present on, an eUICC. Each Profile, combined with the functionality of the eUICC, behaves basically as a SIM card. An eUICC may contain more than one Profile, but one and only one is activated at a time. Each Profile is controlled by a unique ISD-P; consequently, there is one and only one enabled ISD-P at a time on the eUICC.

A Profile can have several forms:

- A Provisioning Profile: A Profile that allows connectivity to a mobile network solely to provide the provisioning of Profiles.
- An Operational Profile: A Profile that allows connectivity to a mobile network.
- A Test Profile: A Profile that can only be used in Device Test Mode and cannot be used to connect to any MNO. The support of this kind of profile is not mandatory for an eUICC implementation.

This ST will use the term "Profile" to describe either Provisioning Profiles, Operational Profiles, or Test Profiles.

All Profiles include Network Access Applications and associated Parameters, but these applications rely on the algorithms stored in the platform layer of the eUICC.

In the same manner, the Profile includes policy rules (PPR), but relies on the Platform Layer to have them enforced on the eUICC. The Profile structure, composed of a set of Profile Components, is specified by, and under the full control of, the MNO. The full Profile structure shall be contained in a unique ISD-P. The Profile structure shall contain a Profile Component, called MNO-SD, which performs an identical Role as the ISD for a UICC. The Profile structure shall include:

- The MNO-SD.
- Supplementary Security Domains (SSD) and a CASD.
- Applets.
- NAAs.
- Other elements of the File System.
- Profile metadata, including Profile Policy Rules (PPR).

More details on the Profile can be found in [23] and [24].

In addition to Profile data, the eUICC itself has a Rules Authorization Table (RAT) that is used by the Profile Policy Enabler (PPE) to determine whether a Profile containing PPRs is authorized and can be installed on the eUICC.

The RAT is initialized at eUICC manufacturing time, or during the initial Device setup provided that there is no installed Operational Profile. It cannot be affected by the Memory Reset function.

ISD-P

The ISD-P is a secure container (Security Domain) for the hosting of a Profile. The ISD-P is also used for updating the Profile Metadata on behalf of the MNO.

As defined in [24], the ISD-P shall ensure that:

a) It hosts a unique Profile.

b) Only the following Application Layer components shall have access to the profiles:

  - ISD-P;
  - ISD-R, which shall only have access to the metadata of the profiles.

c) A Profile component shall not have any visibility of, or access to, components outside its ISD-P. An ISD-P shall not have any visibility of, or access to, any other ISD-P.

d) Deletion of a Profile shall remove the containing ISD-P and all Profile components of the Profile.

ISD-R

The ISD-R is responsible for the creation of new ISD-Ps and life cycle management of all ISD- Ps. An ISD-R shall be created within an eUICC at the time of manufacture.

The ISD-R is used for the Profile download and installation, in collaboration with the Profile Package Interpreter for the decoding/interpretation of the received Profile Package, and with an ISD-P as a target.

As defined in [24]:

a) There shall be only one ISD-R on an eUICC.

b) The ISD-R shall be installed and personalized by the EUM during eUICC manufacturing. The ISD-R shall be associated with itself.

c) The ISD-R cannot be deleted or disabled.

MNO-SD

The MNO-SD is the on-card representative of the MNO Platform. It contains the MNO Over- The-Air (OTA) keys and provides a secure OTA channel.

ECASD

The Embedded UICC Controlling Authority Security Domain (ECASD) is responsible for the secure storage of credentials used to enforce trust in the identities of Actors (eUICC, remote Actors such as SM-DS or SM-DP+) and provides security functions used during key establishment and lpae is pa authentication.

The ECASD is the representative of the off-card entity CI root.

As defined in [23], the ECASD has the following properties:

a) There can only be one ECASD on an eUICC.

b) It is installed and personalized by the EUM during the eUICC manufacturing as described in [11].

c) It has eUICC private key(s) for creating signatures.

d) It has associated certificate(s) for eUICC authentication.

e) It has the Certificate Issuers' (CI) root public key(s) for verifying SM-DP+ and SM-DS certificates.

f) It has the certificate of the EUM.

5G SUCI

This 5G SUCI Applet for Dakota implements SUPI concealment (SUCI) defined by 3GPP TS 33.501 V15.0.0 [44]. The SUCI is a privacy preserving identifier containing the concealed SUPI.

The use of ECIES (Elliptic Curve Integrated Encryption Scheme) for concealment of the SUPI will adhere to SECG specifications [45] and [46].

The function is used in 5GS in the specific cases described in 3GPP TS 33.501 prior to mutual authentication between the UE (User Equipment) and the SN (Serving Network).

The ECIES scheme is implemented to compute a fresh SUCI. USIM will use the provisioned public key of home network and freshly generated ECC (elliptic curve cryptography) ephemeral public/private key pair according to the ECIES parameters provisioned by home network. The processing on UE side shall be done according to the encryption operation defined in [44].

Improved protection of device identity 'over-the-air' includes protection against false base stations. 5G networks use a combination of 'SUPI', a Subscription Permanent Identifier, and 'SUCI' a Subscription Concealed Identity to manage identity of devices or users. This combination provides privacy-preserving protection of device and user identity, ensuring that the real identity cannot be stolen.

Information security will be enhanced in 5G including the implementation of the 'Subscription Concealed Identifier' (SUCI) to encrypt the subscriber identity number (which is part of the international mobile subscriber identify or 'IMSI'). Authentication and encryption protocols ensure that sender and receiver have an established trust and the end-to end relationship is secured. The authentication and encryption is designed to prevent some of the better-known threats such as 'IMSI catchers' and 'man-in-the-middle attacks and more extensive encryption and authentication throughout the networks.

BIPLink

BIPLink is out of scope of the evaluation.

The Bearer Independent Protocol (BIP) is a mechanism by which a mobile phone provides a (U)SIM with access to the data bearers supported by the mobile phone (e.g. Bluetooth, IrDA, etc.) and the network (e.g. GPRS, 3G, etc.).

BIPLink applet is intended to manage the BIP communication required by HTTPs based on GP Amendment B. Therefore, the parameters, process flow and actors involved in the BIP communication managed by BIPLink are closely related with Global Platform and Amendment B Specifications.

In addition to BIP communication, BIPLink applet also handles additional functionalities such as DNS and Load Balancing.

In general, the BIPLink functionalities consist of:

1. BIP Management for GP Amendment B
2. DNS Resolution
3. Load Balancing (only used when searching the first OTA IP list address)
4. Proprietary BIP Event Dispatcher

This protocol application does not embed security functions. This application uses secure channel and secure packets of Platform Layer.

2.4.1.2 Platform layer

The Platform capabilities include:

- The Telecom Framework, which includes algorithms used by Network Access Applications (NAA) to access mobile networks. The NAAs are part of the Profiles, but the algorithms, as part of the Telecom Framework, are provisioned onto the eUICC during manufacturing.
- The Profile Package Interpreter, an eUICC Operating System service that translates the Profile Package data as defined in SIMalliance eUICC Profile Package Specification [53] into an installed Profile using the specific internal format of the target eUICC.
- The Profile Policy Enabler, which has two functions:
    - Verification that a Profile containing PPRs is authorized by the RAT.
    - Enforcement of the PPRs of a Profile.

The Platform Support Function (PSF) embeds Profile management functions in the SDs, the policy enforcement may be realized completely by the ISD-R. The Profile Package Interpreter and Profile Policy Enabler are only defined here to identify the platform code supporting the SDs.

Application Note 1: Authentication to a Public Mobile Network (PMN) is done in accordance with the 3GPP standards [22]. According to these standards (especially TS 33.102) the 3G and 4G authentication mechanisms allow the response values RES to have a length that is any multiple of 8 bits between 32 and 128 bits inclusive. In practice, either 32-bit or 64-bit RES is used. This Security Target covers products only when used to create 64-bit RES. Operators choosing to use 32-bit RES will therefore be using the product outside the scope of this Security Target.

The Security Target includes origin authentication of the PMN that owns the customer subscription to the Profile. It includes also entity authentication of the Profile to the PMN in which a customer subscriber is roaming on. It does not include entity authentication of this visited PMN to the Profile, except in 4G authentication.

The main features addressed by **GP** are:

- The extended GP OPEN for GSMA functions (Profile data management, NAA parameters and policy enforcement)
- The authentication of users through secure channels

- The downloading, installation removal, and selection for execution of Java Card applications
- The life cycle management of both the card and the applications
- The sharing of a global common PIN among all the applications installed on the card

The following GP functionalities, at least, are present within the TOE:

- Card content loading
- Extradition
- Asymmetric keys
- DAP support, Mandated DAP support
- DAP calculation with asymmetric cryptography
- Logical channels

- SCP02, SCP03 [5], SCP03t, SCP80 and SCP81
- Support for contact and contactless cards different implicit selection on different interfaces and channels
- Support for Supplementary Security Domains
- Trusted path privileges
- Post-issuance personalisation of Security Domain [5]
- Application personalisation [5]
- Crypto algorithms as detailed in 1.8.2.2 Cryptographic features

The TOE relies on an **Operating System (OS)** which is an embedded piece of software loaded into the Security IC. The Operating System manages the features and resources provided by the underneath chip. It is, generally divided into two levels:

1) Low level:
   a) Drivers related to the I/O, RAM, SOLID Flash, and any other hardware component present on the Security IC
2) High level:
   a) Protocols and handlers to manage I/O
   b) Memory and file manager
   c) Cryptographic services and any other high level services provided by the OS

The BIOS is an interface between hardware and native components like VM and APIs. The BIOS implements the following functionalities:

- APDU management, using T=0 protocol.
- Timer management
- Exceptions management
- Transaction management
- Flash memory access

2.4.1.3 Cryptographic features

The following crypto services are included in the OS:

Cryptographic Services	Comments
ECC with 192, 256, 384, 512 and 521-bits key sizes	
TDES with 56, 112 and 168-bits key sizes	
AES with 128, 192, 256 key sizes	
SHA-1, SHA 224, 256, 384 and 512 (for data integrity only does not provide confidentiality), SHA-3.	
ECC Key generation	
CRC 16, 32 (for data integrity only does not provide confidentiality)	
RNG CTR_DRBG SP800-90, HASH_DRBG FIPS 186-2	

Cryptographic Services	Comments
ECDSA signature/verification	Based on supported ECC key sizes
ECDH	
HMAC (64 bits up to 1016 bits)	

2.4.1.4 Virtual Machine

The Virtual Machine, which is compliant with the Java Card 3.1 classic edition, interprets the byte code of Java Card applets.
The Virtual Machine supports logical channels; this means that it allows an applet to be selected on a channel, while a different applet is selected on another channel.
It also supports secure execution of applets loaded and stored in FLASH.
The Virtual Machine is activated upon the selection of an applet.

2.4.1.5 The Java Card Runtime Environment

The Java Card Runtime Environment (JCRE) contains the Java Card Virtual Machine (VM), the Java Card Application Programming Interface (API) classes and industry-specific extensions, and support services. For details please refer to reference [29].

2.4.1.6 APIs

The APIs, compliant with the Java Card 3.1 classic edition, support key generation, Key Agreement, signature, ciphering of messages and proprietary IDEMIA API.
Proprietary APIs have been developed like utilBER_Reader to read BER-TLV or Telekom API to manage GSMA communication.

2.4.1.7 Open and isolating Platform

This security target claims conformance to the Application Note 10 on Open and Isolating platform, issued by ANSSI [43].
An "open platform" can host new applications:

- Before its delivery to the end user (during phases 4, 5 or 6 of the traditional smartcard lifecycle). Such loadings are called "pre-issuance".
- After its delivery to the end user (phase 7). Such loadings are called "post-issuance".

An "isolating platform" is a platform that maintains the separation of the execution domains of all embedded applications on a platform, as of the platform itself. "Isolation" refers here to domain separation of applications as well as protection of application's data.

2.4.1.8 Framework Application

The Framework application is the native resident application, with a basic main dispatcher, to receive the card commands and dispatch them to the application and module functions to implement the application commands.
It should not be confused with the default UICC Framework Application.
It also deals with the Card Manufacturer authentication.
The dispatcher is always activated. Commands for administration are only available during pre-personalisation phase.

2.4.1.9 Applets

Applets bytecodes shall go thru the latest Oracle verifier before the loading.
The platform evaluation shall identify, if any, recommendations in order to maintain isolation properties.
These recommendations then shall be followed by the applet developer and shall be checked before loading.

2.4.1.10 JPatch

The platform allows to load patches at pre-personalisation, personalisation or use phase. The patches installed cannot be bypassed. The TOE identification is updated to take into account the patches installed.
The loading of any patch shall follow the procedure of impact analysis defined in [JPATCH].
If the patch reconsiders the security of the TOE, a reassessment of the TOE is mandatory, otherwise a maintenance process is used. The term patch is used in the TSF definitions and refer to JPatch mechanism also called JCVM patch.

2.4.2 Physical scope of the TOE

The physical scope of the TOE is SLI37-SLM37 hardware with loaded IDEMIA software.

The guidance, part of the TOE is defined in table 3.

The guidance is delivered in electronic format with secure transfer.

The IC in delivered in form of diced wafer. The eUICC OS software is embedded in the IC.

The physical part of the TOE (IC+ eUICC) can be delivered in one of the three physical form factors : MFF2, Industrial 2FF and SimFit.

2.5 Life-Cycle

The following description (next table) introduces generics but fine-grained options for the life-cycle of secure products. These options are compliant to standard smartcard life-cycle as defined in [1], [27] and [2]. This document focuses on the eUICC and Java Card platform (the TOE) life cycle which is part of the smart card product life cycle. The intent of the more fine-grained options is to cover the specific aspects of new technologies like platform loading in a comprehensive way and to add some flexibility with respect to the separation of responsibilities between the various parties involved. The product life-cycle is decomposed in phases that describe the competent authorities for each of these phases.

Phase PP0084	Phase eUICC	eUICC Phase name	Actors
1 and 2	a	eUICC platform development: development of IC and Embedded software	Embedded Software : IDEMIA R&D (Jakarta, Courbevoie and Pessac) IC: Infineon
3 and 4	b	eUICC platform storage, pre-perso, tests – Security IC manufacturing and packaging	Infineon Packaging: Infineon or another agent
5	c	eUICC platefrom storage, pre-perso,,test. Platform Loading (using IC Package 1) Integration of Platform Software, applications and pre-perso data.	IDEMIA plant (Haarlem, Vitre, Ostrava, Shenzhen and Noida-P)
6	d	eUICC Personalisation	IDEMIA plant (Haarlem, Vitre, Ostrava, Shenzhen and Noida)
7	e	Operational Usage	The end user

TOE Delivery

Table 4: TOE Life cycle

Notes:

- Notice that the IC loader shall be locked during the pre-personalisation and personalization phase; i.e. before the end-user delivery.
- For a same phase there is no differentiation according activities for IDEMIA sites.

2.5.1 Phase a

2.5.1.1 Security IC Embedded Software development

The platform Development is performed during Phase a. This includes Java Card System (JCS) conception, design, implementation, testing and documentation. The development fulfilled requirements of the final product, including conformance to Java Card Specifications, and recommendations of the user guidance. The development is made in a controlled environment that avoids disclosure of source code, data and any critical documentation and that guarantees the integrity of these elements. The evaluation of the TOE includes the platform development environment.

The code and the associated data are sent

- To IDEMIA audited sites.

2.5.1.2 Security IC Development

The Composite Product life cycle covers Security IC development which is described in the IC ST identification (see corresponding STLite).

2.5.2 Phase b: Security IC Manufacturing and packaging

The Phase b of the Composite Product life cycle covers the IC production Phase 5: Composite Product Integration where the IC is directly delivered without the OS.

2.5.3 Phase c: Platform Loading

The loading takes place at this phase at only IDEMIA audited sites, the loading is done thanks to package 1 of the IC.

2.5.4 Phase d: Composite Product Personalization

This phase is dedicated to the product personalization prior final use.
eUICC personalization covers the insertion of provisioning Profiles and Operational Profiles onto the eUICC.
<u>The TOE delivery is done at this step.</u>

2.5.5 Phase e: Operational Usage

See preparative and operational guidance.

2.6 Summary of the security problem and features

This section aims to provide contextual information regarding the security features described in this Security Target. This high-level view of the Security Target describes:

- The threat agents;
- The main threat categories;
- The Javacard security features.

2.6.1 Threat agents

The two threat agents considered specifically in this Security Target are:

- An off-card Actor.
- An on-card application.

All two types of agents have a High attack potential.

The off-card Actor may be any Actor using the external interfaces of the eUICC, whether they are intended or not to be used.

The intended interfaces of the eUICC are:

- The interfaces with remote provisioning architecture or MNO (TLS interfaces (version 1.2 or later), OTA interfaces, mobile network).
- The interface with the communication module of the Device, which shall conform to the terminal requirements within [6].

The unintended interfaces of the eUICC are mainly the IC surface as defined in [7] (which may include voltage, electro-magnetism, temperature, and so on).

The on-card application is stored on a MNO Profile and uses the following interfaces:

- APIs:
  - GP API,
  - APIs that may be dependent on the Runtime Environment such as the JavaCard API, SIM API ([15]), UICC API ([16]), USIM API ([17]), ISIM API ([18]));

- Policy enforcement interfaces (PPE, PPI);
- APDU buffer / global byte array;
- RE interfaces such as Java Card VM and Java Card RE.

An application may also try to compromise the TOE by directly using an unintended interface such as:

- eUICC memory (via a buffer overflow);
- Access to APDU buffer or global byte array when another application is selected.

This application may also be described as a "malicious on-card application" or "malicious application" in the remainder of this document.

The Platform code itself is not considered a threat agent, since

- Either the runtime environment will be previously certified according to [1];
- Or the runtime environment will be part of the TOE.

In both cases, the IC and its embedded software will be previously certified according to PP0084 [2].

2.6.2 *High-level view of threats*

The threats considered in this Security Target correspond to the high-level scenarios described hereafter.

<u>"First-level" threats</u>

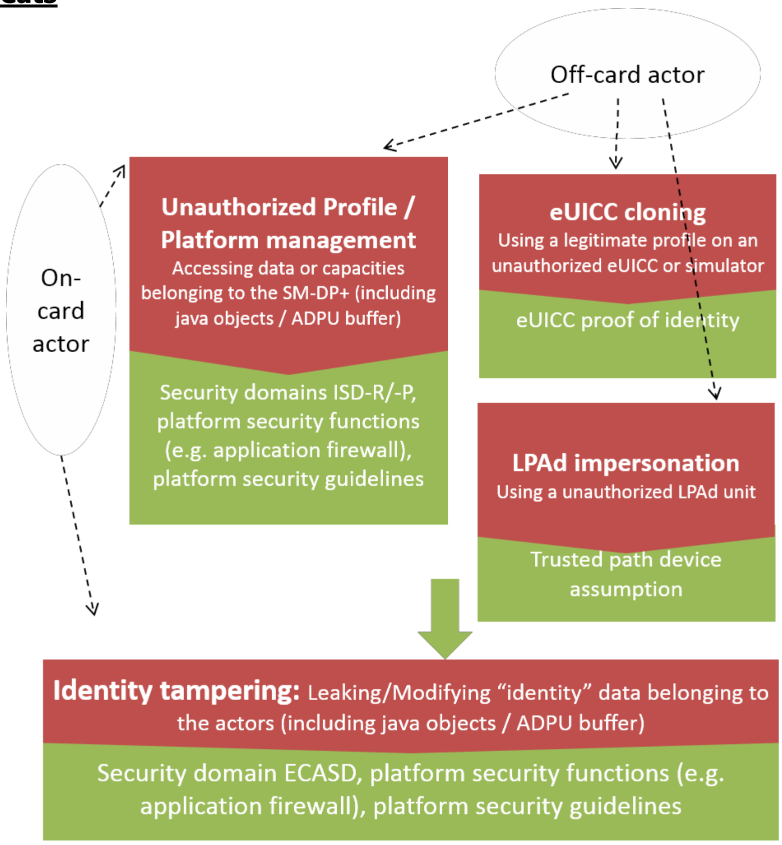

Figure 2: "First-level" threats (1)

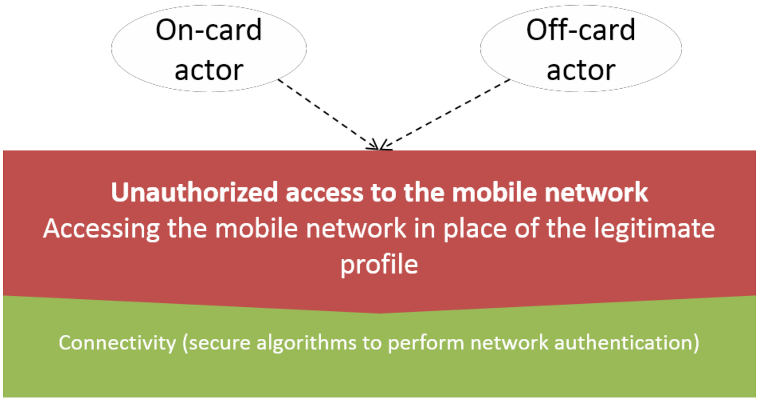

Figure 3: "First-level" threats (2)

2.6.2.1 Unauthorized Profile / Platform management

An off-card Actor or on-card application may try to compromise the eUICC in two different ways, by trying to perform:

- Unauthorized Profile management (typically altering Profile data before or after installation);

- Unauthorized Platform management (typically trying to disable an enabled Profile);

This Security target covers these threats by defining Security Domains: data and capabilities associated to a Security Domain are accessible only to its legitimate owner. The Security Domains are supported by the platform functions. Their isolation is also supported by the Application Firewall provided by the Runtime Environment of the TOE.

The security domain related to the Profile management is the ISD-P, while the security domain in charge of Platform management is the ISD-R.

2.6.2.2 Identity tampering

An attacker may try to bypass the protections against the two categories of threats defined above. A possible vector would consist in directly modifying the identity of the eUICC, or identities of actors via an on-card application. This may be performed, for example, by modifying secrets generated for session establishment, or modifying the CI root public key.

The security objectives covering this threat consist in defining a dedicated Security Domain (ECASD). Identity data such as the CI root public key is under the control of the ECASD and cannot be modified by other actors of the TOE. Some capabilities of the ECASD (such as the generation of secrets) can be used by ISD-R.

The ECASD is supported by the platform functions. Its isolation is also supported by the Application Firewall provided by the Runtime Environment of the TOE.

2.6.2.3 eUICC cloning

An off-card Actor may also try to use a legitimate Profile on an unauthorized eUICC, or on a simulator. The ST prevents cloning by guaranteeing the identity of the eUICC to an off-card Actor before a Profile can be downloaded, or during the usage of the eUICC. The objects used to prove the eUICC identity are controlled by the ECASD security domain.

2.6.2.4 Unauthorized access to the mobile network

An Actor may try to leverage upon flaws of the network authentication algorithms to gain access to network authentication keys, in order to later authenticate in place of a legitimate Profile.

"Second-level" threats

An attacker may try to bypass the protections against the "first-level threats" described in previous section. This ST describes this as "second-level" threats.

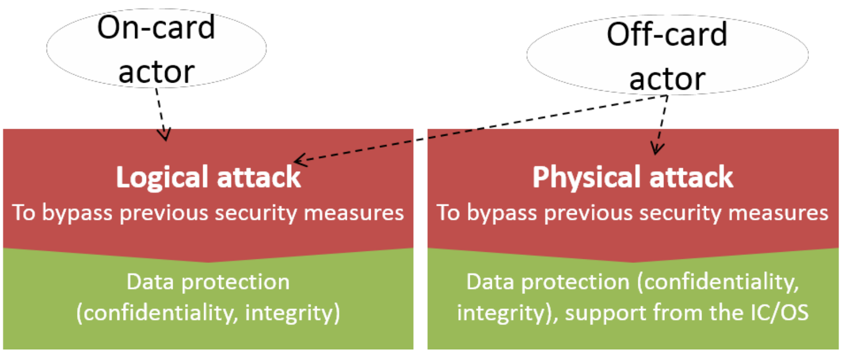

Figure 4: "Second-level" threats

2.6.2.5 Logical attacks

An on-card malicious application, or an off-card Actor, may try to use unintended side-effects of legitimate eUICC functions or commands to bypass the protections of the TSF. This Security Target covers these threats in two different ways:

- The underlying RE protects the Security Domains within the TOE (ISD-R, ISD-P, ECASD) from other applications;
- The Platform code belonging to the TOE is not protected from applications by the RE, thus requiring explicit security objectives;

2.6.2.6 Physical attacks

An off-card Actor may try to bypass the platform TOE functions by several types of attacks. Typically, the off-card Actor may try to perform a side-channel analysis to leak the protected keys, or perform a fault injection to alter the behaviour of the TOE. This Security Target includes security objectives for the underlying IC, which ensures protection against physical attacks.

2.6.3 Javacard security feature

The main goal of the TOE is to provide a sound and secure execution environment to critical assets that need to be protected against unauthorized disclosure and/or modification.

The TOE with its security function has to protect itself and protect applets from bypassing, abuse or tampering of its services that could compromise the security of all sensitive data. Even if the applets are not in the scope of this evaluation.

Atomic Transactions
The TOE shall provide a transaction mechanism. It shall execute a sequence of modifications and allocations on the persistent memory so that either all of them are completed, or the TOE behaves as if none of them had been attempted.
The transaction mechanism shall permit to update internal TSF data as well as to perform different functions of the TOE, like installing a new package on the card.
This mechanism shall be available for applet instances
The TOE shall perform the necessary actions to roll back to a safe state upon interruption.

Card Content Management
The TOE shall control the loading, installation, and deletion of packages and applet instances.
To remove the code of a package from the card, or to definitely deactivate an applet instance, so that it becomes no longer selectable; it shall perform physical removal of those packages and applet data stored in memories (except applet including in OS package in Flash memory that shall only be logically removed).

Card Management Environment
This function shall initialize and manage the internal data structure of the Card Manager. During the initialization phase of the card, it creates the Installer and the Applet Deletion Manager and initializes their internal data structures. The internal data structure of the Card Manager includes the Package and Applet Registries, which respectively contains the currently loaded packages and the currently installed applet instances, together with their associated AIDs.
This function shall also be in charge of dispatching the APDU commands to the applet instances installed on the card and keeping trace of the currently active ones.
It therefore handles sensitive TSF data of other security functions, like the Firewall.

Clearing of sensitive information
The TOE shall ensure that no residual information is available from memories, and shall protect sensitive information that is no longer used. The Platform has to securely clear and destroy this information. It concerns PINs, keys, sensitive data and buffer APDU.
This function is also available to applet.

DAP Verification
An Application Provider may require that its Application code to be loaded on the card shall be checked for integrity and authenticity. The DAP Verification privilege of the Application Provider's Security Domain shall provide this service on behalf of the Application Provider. A Controlling Authority may require that all Application code to be loaded onto the card shall be checked for integrity and authenticity. The Mandated DAP Verification privilege of the Controlling Authority's Security Domain shall provide this service on behalf of the Controlling Authority.

Data coherency
As coherency of data should be maintained, and as power is provided by the CAD and might be stopped at all moment (by tearing or attacks), a transaction mechanism need to be implemented.
When updating data, before writing the new ones, the old ones are saved in a specific memory area.
If a failure appears, at the next start-up, if old data are valid in the transaction area, the system restores them for staying in a coherent state.

Data integrity
Sensitive data have to be protected from modifications: keys, pins, patch code and sensitive applet data.

Encryption and Decryption
The TOE provides the applet instances with a mechanism for encrypting and decrypting the contents of a byte array.
Ciphering operations are implemented to resist environmental stress and glitches and include measures for preventing information leakage through covert channels.

Entity authentication/secure Channel
Off-card entity authentication is achieved through the process of initiating a Secure Channel and provides assurance to the card that it is communicating with an authenticated off-card entity.
If any step in the off-card authentication process fails, the process shall be restarted (i.e. new session keys generated).
The Secure Channel initiation and off-card entity authentication implies the creation of session keys derived from card static key(s).

Exception
In case of abnormal event: data unavailable on an allocation or illegal access to a data, the system shall own an internal mechanism allowing it to stop the code execution and raise an exception.

Firewall
The TOE with the Firewall shall control information flow at runtime. It shall ensure controls object sharing between different applet instances, and between applet instances and the Java Card RE.

Hardware operating
The TOE shall boot after the IC has successfully powered-up. The TOE boot operations shall ensure the correct initialization of the TOE functionalities and the integrity of the code and data.
The TOE shall monitor IC detectors (e.g. out-of-range voltage, temperature, frequency, active shield, memory aging) and shall provide automatic answers to potential security violations through interruption routines that leave the device in a secure state.

Key Access
The TOE shall enforce secure access to all cryptographic keys on the card: DES keys, EC keys, AES keys

Key Agreement
The TOE shall provide to applet instances a mechanism for supporting key agreement algorithms such as EC Diffie-Hellman.

Key destruction
The TOE shall provide secure key destruction, such as keys cannot be retrieved from erased data.

Key Distribution
The TOE shall enforce the distribution of all the cryptographic keys of the card using a specific method.

Key Generation
The TOE shall enforce the creation and the on card generation of all the cryptographic keys of the card using a specific method.

Key management
The TOE shall manage key set: Loading keys, adding a new key set (version and value of the key) or updating a key set (update key value).

Memory failure
This security functionality is in charge of the management of bad usage of the memory.

JPatch at use phase
The loading functionality of patches is also available in use phase, once installed the TOE identification shall take into account the patches installed after delivery.

Random Number
This TOE functionality provides the card manager, the framework application and the applets a mechanism for generating challenges and key values.
The Number Generator is a combination of hardware and software RNG. The RNG is compliant with [48].

3 Conformance Claims

3.1 CC Conformance

This Security Target claims conformance to **[8]**, **[9], and [10]**.

The conformance to the Common Criteria is claimed as follows:

CC	Conformance rationale
Part 2	Conformance to the extended part. FCS.RNG.1: “Random number generation” FIA_API.1: “Authentication Proof of Identity” FPT_EMS.1: “TOE Emanation”
Part 3	Conformance to EAL 4, augmented with AVA_VAN.5: “Advanced methodical vulnerability analysis” ALC_DVS.2: “Sufficiency of security measures” ALC_FLR.1 Basic flaw remediation

Table 5: CC Conformance

Remark

All activities for ALC_FLR.3 have been processed, however, this assurance package is not properly claimed in the present security target as the chip supports only ALC_FLR.1.

3.2 PP Conformance

This security target claims a **demonstrable conformance** to the following Protection Profiles:

Embedded UICC for Consumer Devices Protection Profile [27].

3.3 Conformance Rationale

3.3.1 TOE Type

The TOE type is the one described in [27], chapter 1.2.5, third scenario with the OS represented by the Javacard system included in this current certification and an IC already certified.

The TOE is based on Java Card Open Platform Protection Profile [1] with optional SENSITIVE ARRAY package.

3.3.2 SPD Statement Consistency

All assets, threats, OSPs from the protection profile are included in the security target. All the assumptions from the protection profile have been added in the security target, except A.DELETION.

A.DELETION has been removed from the security target because the deletion of applets is in the scope of the evaluation, as O.CARD_MANAGEMENT is an objective in this security target.

3.3.3 IC in TOE and Objectives on environment

From Javacard PP [1]:
As the IC is included in the TOE, OE.CARD-MANAGEMENT, OE.SCP.RECOVERY, OE.SCP.SUPPORT, and OE.SCP.IC are changed into the following Objectives on the TOE: O.CARD-MANAGEMENT, O.SCP.RECOVERY, O.SCP.SUPPORT, and O.SCP.IC.
As the SCP is included in the TOE, OE.SCP.RECOVERY, OE.SCP.SUPPORT, and OE.SCP.IC are changed into the following Objectives on the TOE: O.SCP.RECOVERY, O.SCP.SUPPORT, and O.SCP.IC.

From eUICC PP [27]:
The Runtime environment is included in the OS so the objectives OE.RE* are transformed into objectives of the TOE or by re-using the Javacard PP objectives or excluded:

Objective on environment in eUICC PP	Objective in TOE
OE.IC.PROOF_OF_IDENTITY	The IC is uniquely identified according to its certificate: see chapter 2.2
OE.IC.SUPPORT	Equivalent to OE.SCP.SUPPORT of [1] integrated in the TOE by O.SCP. SUPPORT
OE.IC.RECOVERY	Equivalent to OE.SCP.RECOVERY of [1] integrated in the TOE by O.SCP.RECOVERY
OE.RE.PPE-PPI	OE.RE.PPE-PPI is reused by the security objectives of [1] related to the following threats: T.DELETION, T.INSTALL so covered by O.CARD_MANAGEMENT, O.INSTALL, O.DELETION and O.LOAD of the TOE.
OE.RE.SECURE-COMM	Re-use the security objectives of [1] related to the following threats: T.CONFID-APPLI-DATA and T.INTEG-APPLI-DATA
OE.RE.API	Re-use the security objectives of [1] related to the following threats: T.CONFID-JCS-CODE, T.INTEG-JCS-CODE, T.CONFID-JCS-DATA, T.INTEG-JCS-DATA
OE.RE.DATA-CONFIDENTIALITY	Objective translated by o Re-use the security objectives of [1] related to the following threats: T.CONFID-APPLI-DATA; o refining the ADV_ARC "non-bypassability" requirements to explicit the coverage of side channel attacks by the security architecture of the TOE.
OE.RE.DATA-INTEGRITY	Re-use the security objectives of [1] related to the following threats: T.INTEG-APPLI-DATA, T.INTEG-APPLI-DATA.LOAD, T.INTEG-APPLI-CODE, T.INTEG-APPLI-CODE.LOAD
OE.RE.IDENTITY	Re-use O.SID of [1]
OE.RE.CODE-EXE	Re-use the security objectives of [1] related to the following threats: T.EXE-CODE.1, T.EXE-CODE.2, T.EXE-CODE-REMOTE and T.NATIVE.

3.3.4 Assumptions

All the assumptions from the protection profile have been added in the security target, except A.DELETION.

A.DELETION has been removed from the security target because the deletion of applets is in the scope of the evaluation, as O.CARD_MANAGEMENT is an objective in this security target. Other assumptions have been added

3.3.5 Security Objectives

All the security objectives for the TOE from the protection profile are included in the security target. The security objectives on environment dedicated to the Runtime Environment OE.RE. in PP eUICC [27] are defined in security objectives for Javacard as described in section 3.3.3

Other security objectives for the TOE have been added:

Four security objectives for the operational environment defined in the PP JCS [1] have been transformed in security objectives for the TOE:

- OE.SCP.IC → **O.SCP.IC**
- OE.SCP.SUPPORT → **O.SCP.SUPPORT**
- OE.SCP.RECOVERY → **O.SCP.RECOVERY**
- OE.CARD_MANAGEMENT → **O.CARD_MANAGEMENT**

One security objectives for the operational environment defined in the PP SGP.25 [27] have been transformed in security objectives for the TOE:

- OE.RE.PPE-PPI is reused by the security objectives of [1] related to the following threats: T.DELETION, T.INSTALL. It implies so O.CARD_MANAGEMENT (O.DELETION, O.INSTALL and O.LOAD here included in O.CARD_MANAGEMENT).

The Objective **O.SENSITIVE_ARRAYS_INTEG** has been added from additional package SENSITIVE ARRAY in PP Javacard [1].

Also the following objectives have been added:

- **O.PATCH_LOADING** for the secure patch code loading mechanism.
- **O.RESOURCES** for stack overflow managing defined in PP [1].
- **O.OBJ-DELETION** for secure object deletion defined in PP [1].

3.3.6 Security Functional Requirements

All SFRs from the protection profiles [1] and [27] have been added in the security target. Other SFRs have been added to cover supplemental security objectives: see the rational.

PP EUiCC [27]:
Also, the following SFRs have been renamed to avoid duplication with the SFRs defined in the Java Card PP [1]:

SFR from PP [27]	ST SFR New name
FMT_SMF.1	FMT_SMF.1/EUICC
FMT_SMR.1	FMT_SMR.1/EUICC
FPT_FLS.1	FPT_FLS.1/EUICC

The definition of Extended Components comes from [27], the concerned SFRs are:

- FPT_EMS.1 TOE Emanation
- FIA_API - Authentication Proof of Identity
- FCS_RNG - Random number generation

PP Javacard [1]:
The following SFRs from PP Javacard [1] have been renamed to avoid duplication with the SFRs defined in the eUICC PP **[27]**:

SFR from PP [1]	ST SFR New name
FCS_COP.1	FCS_COP.1/Disp

The optional package SENSITIVE ARRAY has been added.
These following SFR are also added for javacard functionalities:
Iterations have been added for runtime, linked to O.Firewall and used for Stack, Transient and Heap, FPT_TST.1 is added to reflect selftests.

Requirements added for Javacard PP [1]	Rational
FPT_TST.1	added for reflecting selftests from [9].
FCO_NRO.2/CM_DAP	added to manage the DAP from [9].
FIA_UAU.1/CM	Iteration Added to enlarge FIA_UAU.1/EXT from [9].
FIA_UAU.4/CardIssuer	Iteration Added to enlarge FIA_UAU.4/EXT from [9].
FPT_TDC.1/CM	Iteration for the capacity to interpret the keyset from [1].
FCS_COP.1/CM-SCP	Iteration to express the secure channel protocols from [1].
FDP_ACC.2/Patch	Used for patch loading, added from [9].
FDP_ACF.1/Patch	Used for patch loading, added from [9].
FDP_UCT.1/Patch	Used for patch and locks, added from [9].
FDP_ITC.1/Patch	Used for patch loading, added from [9].
FCS_COP.1/Patch	Used for patch loading, added from [9].
FDP_UIT.1/Patch	Used for patch loading, added from [9].
FAU_STG.2/Patch	Used for patch identification, added from [9].
FPT_RCV.4/SCP	Added to cover O.IC.SUPPORT from [1].
FDP_ACC.2/RV_Stack	Added for stack security from [1].
FDP_ACF.1/RV_Stack	Added for stack security from [1].
FMT_MSA.1/RV_Stack	Added for stack security from [1].
FMT_MSA.2/RV_Stack	Added for stack security from [1].
FMT_MSA.3/RV_Stack	Added for stack security from [1].
FMT_SMF.1/RV_Stack	Added for stack security from [1].
FDP_ACC.2/RV_Heap	Added for heap security from [1].
FDP_ACF.1/RV_Heap	Added for heap security from [1].
FMT_MSA.1/RV_Heap	Added for heap security from [1].
FMT_MSA.2/RV_Heap	Added for heap security from [1].
FMT_MSA.3/RV_Heap	Added for heap security from [1].
FMT_SMF.1/RV_Heap	Added for heap security from [1].
FDP_ACC.2/RV_Transient	Added for transient security management from [1].
FDP_ACF.1/RV_Transient	Added for transient security management from [1].
FMT_MSA.1/RV_Transient	Added for transient security management from [1].
FMT_MSA.2/RV_Transient	Added for transient security management from [1].
FMT_MSA.3/RV_Transient	Added for transient security management from [1].
FMT_SMF.1/RV_Transient	Added for transient security management from [1].

SUCI APPLET
The following SFR has been added for 5G SUCI Applet:

Requirements added for Javacard PP [1]	Rational
FCS_CKM.1/SUCI	Added for 5G SUCI applet from [9]
FCS_CKM.4/SUCI	Added for 5G SUCI applet from [9]
FCS_COP.1/SUCI	Added for 5G SUCI applet from [9]

3.3.7 SAR

This ST claims EAL4 augmented with ALC_DVS.2 and AVA_VAN.5 as defined in PP eUICC [27], ALC_FLR.1 is added.

4 Security aspects

This chapter describes the main security issues of the Java Card System and its environment addressed. They can be instantiated as assumptions, threats, objectives (for the TOE and the environment) or organizational security policies.

For instance, we will define hereafter the following aspect:

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

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

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

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.

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.

#.INTEG-APPLI-DATA-PHYS:
Integrity-sensitive application data must be protected against unauthorized modification by physical attacks.

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;.

#.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 not read, write, compare a piece of data belonging to an applet that is not in the same context, or execute one of the methods of an applet in another context without its authorization.

#.NATIVE:
Because the execution of native code is outside of the JCS TSF scope, it must be secured so as to not provide ways to bypass the TSFs of the JCS. Loading of native code, which is as well outside those TSFs, is submitted to the same requirements. Should native software be privileged in this respect, exceptions to the policies must include a rationale for the new security framework they introduce.

4.4 Bytecode verification

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

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 ([8], [41], [1]). The actual set of checks performed by the verifier is implementation-dependent, but shall at least enforce all the "must clauses" imposed in [8] 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 [8] §4.4), 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 into 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 authentication

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 IC issuer's policy on the card or IC.

#.INSTALL:
(1) The TOE must be able to return to a safe and consistent state when the installation of a package or an applet fails or be cancelled (whatever the reasons). (2) Installing an applet must have no effect on the code and data of already installed applets. The installation procedure should not be used to bypass the TSFs. In short, it is an atomic operation, free of harmful effects on the state of the other applets. (3) The procedure of loading and installing a package shall ensure its integrity and authenticity.

#.SID:
(1) Users and subjects of the TOE must be identified. (2) The identity of sensitive users and subjects associated with administrative and privileged roles must be particularly protected; this concerns the Java Card RE, the applets registered on the card, and especially the default applet and the currently selected applet (and all other active applets in Java Card System 2.2.x). A change of identity, especially standing for an administrative role (like an applet impersonating the Java Card RE), is a severe violation of the Security Functional Requirements (SFR). Selection controls the access to any data exchange between the TOE and the CAD and therefore, must be protected as well. The loading of a package or any exchange of data through the APDU buffer (which can be accessed by any applet) can lead to disclosure of keys, application code or data, and so on.

#OBJ-DELETION:
(1) Deallocation of objects should not introduce security holes in the form of references pointing to memory zones that are not longer in use, or have been reused for other purposes. Deletion of collection of objects should not be maliciously used to circumvent the TSFs. (2) Erasure, if deemed successful, shall ensure that the deleted class instance is no longer accessible.

#DELETION:
(1) Deletion of installed applets (or packages) should not introduce security holes in the form of broken references to garbage collected code or data, nor should they alter integrity or confidentiality of remaining applets. The deletion procedure should not be maliciously used to bypass the TSFs. (2) Erasure, if deemed successful, shall ensure that any data owned by the deleted applet is no longer accessible (shared objects shall either prevent deletion or be made inaccessible). A deleted applet cannot be selected or receive APDU commands. Package deletion shall make the code of the package no longer available for execution. (3) Power failure or other failures during the process shall be taken into account in the implementation so as to preserve the SFRs. This does not mandate, however, the process to be atomic. For instance, an interrupted deletion may result in the loss of user data, as long as it does not violate the SFRs.
The deletion procedure and its characteristics (whether deletion is either physical or logical, what happens if the deleted application was the default applet, the order to be observed on the deletion steps) are implementation-dependent. The only commitment is that deletion shall not jeopardize the TOE (or its assets) in case of failure (such as power shortage).
Deletion of a single applet instance and deletion of a whole package are functionally different operations and may obey different security rules. For instance, specific packages can be declared to be undeletable (for instance, the Java Card API packages), or the dependency between installed packages may forbid the deletion (like a package using super classes or super interfaces declared in another package).

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, itis required that (7) the IC is designed in accordance with a well-defined set of policies and standards (for instance, those specified in [24]), and will be tamper resistant to actually prevent an attacker from extracting or altering security data (like cryptographic keys) by using commonly employed techniques (physical probing and sophisticated analysis of the chip). This especially matters to the management (storage and operation) of cryptographic keys.

#.TRANSACTION:
The TOE must provide a means to execute a set of operations atomically. This mechanism must not jeopardise the execution of the user applications. The transaction status at the beginning of an applet session must be closed (no pending updates).

5 Security Problem Definition

5.1 Assets

Assets are security-relevant elements to be directly protected by the TOE. For each asset it is specified the kind of risks they run.

5.1.1 Java Card

5.1.1.1 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, array view, 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, an array view 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.

5.1.1.2 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 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.

S.CAP_FILE

A CAP file may contain multiple Java language packages. A package is a namespace within the Java programming language that may contain classes and interfaces. A CAP file may contain packages that define either user library, or one or several applets. A CAP file compliant with Java Card Specifications version 3.1 may contain multiple Java language packages. An EXTENDED CAP file as specified in Java Card Specifications version 3.1 may contain only applet packages, only library packages or a combination of library packages. A COMPACT CAP file as specified in Java Card Specifications version 3.1 or CAP files compliant to previous versions of Java Card Specification, MUST contain only a single package representing a library or one or more applets.

5.1.1.3 Additional assets

D.CONFIG

The patches are loaded into the TOE. These elements of configuration have to be loaded securely. To be protected from unauthorized disclosure or modification.

D.SENSITIVE_DATA

The other sensitive data are grouped in the same D.Sensitive_Data. The list is presented below:

- D.NB_AUTHENTIC: Number of authentications. This number is specified in the SFR
- D.NB_REMAINTRYOWN: Number of remaining tries for owner PIN. This number is specified in the SFR
- D.NB_REMAINTRYGLB: Number of remaining tries for a global PIN. This number is specified in the SFR
- ASG.CARDREG: Card registry (AS.APID: Applet Identifier (AID), AS.CMID: Card Manager ID (AID))
- ASG.APPRIV: Applet privileges group (Card Manager lock privilege, Card terminate privilege, Default selected privilege, PIN change privilege, Security Domain privilege, Security Domain with DAP verification privilege, Security Domain with Mandated DAP verification privilege)
- AS.CMLIFECYC: this Security Attribute represents the Card life cycle state. It can be either: Prepersonalisation, Personalisation and use phases of the card.

D.JCS_KEYS

AS.KEYSET_VERSION and AS.KEYSET_Value Cryptographic keys used when loading a file into the card. To be protected from unauthorized disclosure and modification.

5.1.2 Consumer Device

The Assets in this category are divided into two groups. The first one contains the data created by and for the user (User data) and the second one includes the data created by and for the TOE (TSF data).

5.1.2.1 User Data

User data includes:

- User data controlled by the ISD-P:
  - At least one Network Authentication --Application (part of D.PROFILE_CODE) and its associated parameters (D.PROFILE_NAA_PARAMS);
  - The PPR policy file (D.PROFILE_POLICY_RULES);
  - The file system (included in D.PROFILE_CODE);
  - The MNO-SD, which may include other applications, as well as:
    - The identity associated with the profile (D.PROFILE_IDENTITY),
    - The MNO-SD keyset (D.MNO_KEYS);
  - The user codes that may be associated to the profile download (D.PROFILE_USER_CODES).

Keys

Cryptographic keys owned by the Security Domains. All keys are to be protected from unauthorized disclosure and modification.

D.MNO_KEYS

Keys used by MNO OTA Platform to request management operations from the ISD-P. The keys are loaded during provisioning and stored under the control of the MNO SD.

Profile data

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, including confidential sensitive data.

D.PROFILE_NAA_PARAMS

Parameters used for network authentication, including keys. Such parameters may include for example elliptic curve parameters. Parameters are loaded during provisioning and stored under the control of the ISD-P. They may be transmitted to the Telecom Framework, which contains the authentication algorithms. To be protected from unauthorized disclosure and unauthorized modification.

D.PROFILE_IDENTITY

The International Mobile Subscriber Identity is the user credential when authenticating on a MNO's network via an Authentication algorithm. The IMSI is a representation of the subscriber's identity and will be used by the MNO as an index for the subscriber in its HLR. Each IMSI is stored under the control of the ISD-P during provisioning. The IMSI shall be protected from unauthorized modification.

D.PROFILE_POLICY_RULES

Data describing the profile policy rules (PPRs) of a profile. These rules are loaded during provisioning and stored under the control of the ISD-P. They are managed by the MNO OTA Platform. PPRs shall be protected from unauthorized modification.

D.PROFILE_USER_CODES

This asset consists of:

- the optional Activation Code that End User may use to initiate a Profile Download and Installation via the Local User Interface (LUId);
- the hash of the optional Confirmation Code (Hashed Confirmation Code) that End User may use to confirm a Profile Download and Installation via the Local User Interface (LUId). Note that although these codes are input by End User at the LUId, which is outside of the TOE, the codes are sent to the TOE for signature (ex. euiccSigned2 data structure). To be protected from unauthorized modification.

Profile code

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, including confidential sensitive data.

D.PROFILE_CODE

The profile applications include first and second level applications ([6]), in particular:

- The MNO-SD and the Security Domains under the control of the MNO-SD (CASD, SSD);
- The other applications that may be provisioned within the MNO-SD (network access applications, and so on). This asset also includes, by convention, the file system of the Profile. All these applications are under the control of the MNO SD. These assets have to be protected from unauthorized modification.

5.1.2.2 TSF data

The TSF data includes three categories of data:

- TSF code, ensuring the protection of Profile data;
- Management data, ensuring that the management of applications will enforce a set of rules (for example privileges, life-cycle, and so on);
- Identity management data, guaranteeing the identities of eUICC and remote actors.

TSF Code

D.TSF_CODE

The TSF Code distinguishes between

- the ISD-R, ISD-Ps and ECASD;
- the Platform code. All these assets have to be protected from unauthorized disclosure and modification. Knowledge of this 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 code is stored.

Application Note:

- this does not include applications within the MNO-SD, which are part of the user data (Profile applications);

- the notion of unauthorized disclosure and modification is the same as used in [1].

Management data

D.PLATFORM_DATA

The data of the platform environment, like for instance,

- the identifiers and privileges including SM-DS OID, MNO OID and SM-DP+ OID;
- the eUICC life-cycle state of the ISD-P security domain (see Annex A of [24]). This data may be partially implemented in the logic of ISD-R and the Platform code, instead of being "data" properly speaking. As a consequence, this asset is strongly linked with D.TSF_CODE. To be protected from unauthorized modification.

D.DEVICE_INFO

This asset includes the security-sensitive elements of Device Information data, such as the device type allocation code (TAC) or the device capabilities. To be protected from unauthorized modification.

D.PLATFORM_RAT

Data describing the Rules Authorisation Table (RAT) of the eUICC. These rules are initialised at eUICC manufacturing time or during the initial device setup provided that there is no installed operational profile. The OEM or EUM is responsible for setting the content of the RAT. RAT is stored in the eUICC. To be protected from unauthorized modification.

Identity management data

Identity management data is used to guarantee the authenticity of actor's identities. It includes:

- EID, eUICC certificate and associated private key, which are used to guarantee the identity of the eUICC;
- CI's root certificate (self-signed), which is used to verify all actor's certificates;
- EUM's certificates;
- Shared secrets used to generate credentials.

D.SK.EUICC.ECDSA

The eUICC private key(s), stored in ECASD, used by the eUICC to prove its identity and generate shared secrets with remote actors. It must be protected from unauthorized disclosure and modification.

D.CERT.EUICC.ECDSA

Certificate(s) issued by the EUM for a specific, individual, eUICC. Certificates contain public keys PK.EUICC.ECDSA and are stored in ECASD. This certificate(s) can be verified using the EUM Certificate. The eUICC certificate(s) has to be protected from unauthorized modification.

D.PK.CI.ECDSA

The CI's public key (PK.CI.ECDSA) used to verify the certification chain of eUICC and remote actors. It is stored in ECASD. It must be protected from unauthorized modification. ECASD MAY contain several public keys belonging to the same GSMA CI or different GSMA CIs. Each PK.CI.ECDSA SHALL be stored with information coming from the CERT.CI.ECDSA the key is included in, at least:

- Certificate serial number: required to manage GSMA CI revocation by CRL;
- GSMA Certificate Issuer Identifier: GSMA CI OID;

- Subject Key Identifier: required to verify the Certification chain of the off-card entity.

D.EID

The EID (eUICC-ID) uniquely identifies the eUICC. This identifier is set by the eUICC manufacturer and does not change during operational life of the eUICC. It is stored in ECASD. The EID is used as a key by SM-DP+ and SM-DS to identify eUICCs in their databases. The EID shall be protected from unauthorized modification.

D.SECRETS

This asset includes:

- the one-time keys of the eUICC and the SM-DP+: otSK.EUICC.ECKA, otPK.EUICC.ECKA and otPK.DP.ECKA;
- the shared secret (ShS) used to protect the Profile download; and
- session keys (S-ENC and S-MAC) and the initial MAC chaining value. These asset shall be protected from unauthorized disclosure and modification.

D.CERT.EUM.ECDSA

The Certificate(s) of the EUM (CERT.EUM.ECDSA). To be protected from unauthorised modification.

D.CRLs

The optional certificate revocation lists (extract) stored in the eUICC. To be protected against unauthorised modification.

5.2 Users / Subjects

5.2.1 Java Card

5.2.1.1 Additional Users / Subjects

S.FRAMEWORK_APPLICATION

The Framework application

5.2.1.2 Miscellaneous

S.ADEL

The applet deletion manager which also acts on behalf of the card issuer. It may be an applet ([29], §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 ADELG Security Functional Requirements.

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

S.CAD

The CAD represents the actor that requests, by issuing commands to the card. It also plays the role of the off-card entity that communicates with the S.INSTALLER.

S.INSTALLER

The installer is the on-card entity which acts on behalf of the card issuer. This subject is involved in the loading of packages and installation of applets.

S.JCRE

The runtime environment under which Java programs in a smart card are executed.

S.JCVM

The bytecode interpreter that enforces the firewall at runtime.

S.LOCAL

Operands stack of a JCVM frame, or local variable of a JCVM frame containing an object or an array of references.

S.MEMBER

Any object's field, static field or array position.

S.PACKAGE

A package is a namespace within the Java programming language that may contain classes and interfaces, and in the context of Java Card technology, it defines either a user library, or one or several applets.

S.TOE

Source code.

5.2.2 Consumer Device

This section distinguishes between:

- users, which are entities external to the TOE that may access its services or interfaces;
- subjects, which are specific parts of the TOE performing specific operations. The subjects are subparts of the asset D.TSF_CODE.

5.2.2.1 Users

U.SM-DPplus

Role that prepares the Profiles and manages the secure download and installation of these Profiles onto the eUICC.

U.MNO-OTA

An MNO platform for remote management of UICCs and the content of Enabled MNO Profiles on eUICCs.

5.2.2.2 Subjects

S.ISD-R

The ISD-R is responsible for the creation of new ISD-Ps and life-cycle management of all ISD-Ps.

S.ISD-P

The ISD-P is the on-card representative of the SM-DP+ and is a secure container (Security Domain) for the hosting of a Profile.

S.ECASD

The Embedded UICC Controlling Authority Security Domain (ECASD) is responsible for secure storage of credentials required to support the required security domains on the eUICC.

S.PPI

Profile Package Interpreter, an eUICC Operating System service that translates the Profile Package data as defined in SIMalliance eUICC Profile Package Specification [5] into an installed Profile using the specific internal format of the target eUICC.

S.PPE

Profile Policy Enabler, which has two functions:

- Verification that a Profile containing PPRs is authorised by the RAT;
- Enforcement of the PPRs of a Profile.

S.TELECOM

The Telecom Framework is an Operating System service that provides standardised network authentication algorithms to the NAAs hosted in the ISD-Ps.

5.3 Threats

5.3.1 Java Card

5.3.1.1 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, S.CAP_FILE 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, D.CRYPTO, S.CAP_FILE and D.JCS_KEYS.

5.3.1.2 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_KEY.

T.INTEG-JCS-CODE

The attacker executes an application to alter (part of) the Java Card System code. See #.INTEG-JCS-CODE for details.

Directly threatened asset(s): D.JCS_CODE.

T.INTEG-JCS-DATA

The attacker executes an application to alter (part of) Java Card System or API data. See #.INTEG-JCS-DATA for details.

Directly threatened asset(s): D.API_DATA, D.SEC_DATA, D.JCS_DATA, D.JCS_KEYS 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.

5.3.1.3 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, D.JCS_KEYS, D.APP_KEYs and D.SENSITIVE_DATA, S.CAP_FILE.

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), S.CAP_FILE and D.SENSITIVE_DATA.

5.3.1.4 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 TOE Security Function such as the firewall. See #.NATIVE for details.

Directly threatened asset(s): D.JCS_DATA.

5.3.1.5 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.

5.3.1.6 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 for details.

Directly threatened asset(s): D.SEC_DATA, D.APP_CODE and D.SENSITIVE_DATA.

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) and D.SENSITIVE_DATA.

5.3.1.7 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.

5.3.1.8 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 in the present evaluation (restricted to physical attacks).

This threat refers to the point (7) of the security aspect #.SCP, and all aspects related to confidentiality and integrity of code and data.

5.3.1.9 Additional threats

T.PATCH_LOADING

The attacker tries to avoid the loading of a genuine patch by:

- o altering a patch (during loading or once loaded),
- o exploiting the patch loading mechanism to load unauthenticated code on the TOE

in order to get access to the assets, the TSF data or the TOE user data, or to modify the TSF.

Directly threatened asset(s): D.CONFIG

5.3.2 Consumer Device

5.3.2.1 Identity tampering

T.UNAUTHORIZED-IDENTITY-MNG

A malicious on-card application:

- o discloses or modifies data belonging to the "Identity management data" or the "TSF Code" asset category:
  - ▪ discloses or modifies D.SK.EUICC.ECDSA, D.SECRETS,
  - ▪ modifies D.CERT.EUICC.ECDSA, D.PK.CI.ECDSA, D.EID, D.CERT.EUM.ECDSA, D.CRLs,
  - ▪ modifies the generation method (part of D.TSF_CODE) for shared secrets, one-time keys or session keys (i.e. methods used to generate D.SECRETS);
- o discloses or modifies functionalities of the ECASD (part of D.TSF_CODE). Such a threat typically includes for example:
- o direct access to fields or methods of the Java objects
- o exploitation of the APDU buffer and global byte array
- o impersonation of an application, of the Runtime Environment, or modification of privileges of an application

T.IDENTITY-INTERCEPTION

An attacker may try to intercept credentials, either on-card or off-card, in order to

- use them on another eUICC or on a simulator
- modify them / replace them with other credentials. This includes on-card interception of:
- the shared secrets used in profile download (D.SECRETS)
- the eUICC-ID (D.EID) This does not include:
- off-card or on-card interception of SM-DP+ credentials during profile download (taken into account by T.PROFILE-MNG-INTERCEPTION) Directly threatens the assets: D.SECRETS, D.EID.

5.3.2.2 Unauthorized profile and platform management

An off-card actor or on-card application may try to compromise the eUICC by trying to perform:

- Either unauthorized Profile Management (typically accessing or modifying the content of a profile, for example altering a downloaded profile before installation, or leaking the network authentication parameters stored in the profile);
- Or unauthorized Platform Management (typically trying to disable an enabled profile).

T.UNAUTHORIZED-PROFILE-MNG

A malicious on-card application:

- modifies or discloses profile data belonging to ISD-P or MNO-SD;
- executes or modifies operations from profile applications (ISD-P, MNO-SD and applications controlled by MNO-SD);
- modifies or discloses the ISD-P or MNO-SD application. Such threat typically includes for example:
- direct access to fields or methods of the Java objects;
- exploitation of the APDU buffer and global byte array.

The ST does not address the following cases:

- An application within a ISD-P tries to compromise its own MNO-SD;
- An application within a ISD-P tries to compromise another application under the control of its own MNO-SD or ISD-P.

These cases are considered the responsibility of the MNO, since they only compromise their own profile, without any side-effect on other MNO profiles.

The ST addresses the following cases:

- An application within a ISD-P tries to compromise another MNO-SD or ISD-P;
- An application within a ISD-P tries to compromise an application under the control of another MNO-SD or ISD-P;
- An application within a ISD-P tries to compromise its own ISD-P. The first two cases have an impact on other MNO profiles for trivial reasons. The last case would consist, for example, in modifying the fallback attribute of the ISD-P, thus having an impact on the whole Platform Management behaviour.

T.UNAUTHORIZED-PLATFORM-MNG

An on-card application:

- modifies or discloses data of the ISD-R or PPE;
- executes or modifies operations from ISD-R or PPE;
- modifies the rules authorisation table (RAT) stored in the PPE.

Such a threat typically includes for example:

- direct access to fields or methods of the Java objects
- exploitation of the APDU buffer and global byte array

T.PROFILE-MNG-INTERCEPTION

An actor alters or eavesdrops the transmission between eUICC and SM-DP+ (ES8+), or eUICC and MNO OTA Platform (ES6), in order to:

- disclose, replace or modify the content of a profile during its download to the eUICC;
- download a profile on the eUICC without authorization;
- replace or modify the content of a command from SM-DP+ or MNO OTA platform;
- replace or modify the content of Profile Metadata (ex. the Profile Policy Rules (PPR)) data when updated by the MNO OTA platform.

Application Note:

NB: the attacker may be an on-card application intercepting transmissions to the security domains, or an off-card actor intercepting OTA transmissions or interface between the eUICC and the Device.

T.PROFILE-MNG-ELIGIBILITY

An actor alters or eavesdrops the transmission between eUICC and SM-DP+ (ES8+), or alters the Device Information, in order to compromise the eligibility of the eUICC, for example:

- downgrade the security of the profile sent to the eUICC by claiming compliance to a previous version of the specification, or lack of cryptographic support;
- obtain an unauthorized profile by modifying the Device Info or eUICC identifier.

Application Note:

NB: the attacker may be an on-card application intercepting transmissions to the security domains, or an off-card actor intercepting OTA transmissions or interface between the eUICC and the Device.

5.3.2.3 eUICC cloning

T.UNAUTHORIZED-eUICC

The attacker uses a legitimate profile on an unauthorized eUICC, or on any other unauthorized support (for example a simulator or soft SIM).

5.3.2.4 Unauthorized access to the mobile network

T.UNAUTHORIZED-MOBILE-ACCESS

An on-card or off-card actor tries to authenticate on the mobile network of a MNO in place of the legitimate profile.

5.3.2.5 Second level threats

T.PHYSICAL-ATTACK has been removed as it is already covered by T.PHYSICAL of the Java Card

T.LOGICAL-ATTACK

An on-card malicious application bypasses the Platform security measures by logical means, in order to disclose or modify sensitive data when they are processed by the Platform:

- IC and OS software
- Runtime Environment (for example provided by JCS)

- the Profile Policy Enabler
- the Profile Package Interpreter
- the Telecom Framework (accessing Network Authentication Parameters). An example of such a threat would consist of using buffer overflows to access confidential data manipulated by native libraries. This threat also includes cases of unauthorized code execution by applications.

5.3.2.6 LPAd impersonation

T.LPAd INTERFACE EXPLOIT

The attacker exploits the interfaces to LPAd (interfaces ES10a, ES10b and ES10c) to:

- either impersonate the LPAd (Man-in-the-middle, masquerade), or
- exploit a flaw in the interface to modify or disclose sensitive assets, or execute code (extension of T.LOGICAL-ATTACK and T.PHYSICAL-ATTACK targeting specifically the interfaces to LPAd). The attacker could thus perform unauthorised profile and platform management, for instance by circumventing the End User confirmation needed for such actions. The attacker could also compromise the eligibility check process by compromising the Device Information that is normally passed on from the LPA to the eUICC before profile download and installation. The difference to the threats T.UNAUTHORIZED-PROFILE-MNG, T.UNAUTHORIZED-PLATFORM-MNG, and T.PROFILE-MNG-ELIGIBILITY, is on the interfaces used to perform the attack (ES10a,b,c). Directly threatened asset: D.DEVICE_INFO, D.PLATFORM_DATA. Recall that LPAd is an optional and non-TOE component, but even when LPAd is not present, the interfaces to LPAd (ES10a,b,c) are present.

5.4 Organisational Security Policies

5.4.1 Java Card

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. OE.VERIFICATION guarantees the correct integrity and authenticity evidences for each application, by means of elements provided by OE.CODE-EVIDENCE.

5.4.2 Consumer Device

5.4.2.1 Life-cycle

OSP.LIFE-CYCLE

The TOE must enforce the eUICC life-cycle defined in [24]. In particular:

- There is only one ISD-P enabled at a time;
- The eUICC must enforce the profile policy rules (PPR) in case a profile state change is attempted (installation, disabling or deletion of a profile), except during the memory reset or test memory reset functions: in this case, the eUICC may disable and delete the currently enabled profile, even if a PPR states that the profile cannot be disabled or deleted;
- The eUICC must enforce the rules authorisation table (RAT) before a profile containing PPRs is authorised to be installed on the eUICC.

5.5 Assumptions

5.5.1 Java Card

A.CAP_FILE

CAP Files loaded post-issuance do not contain native methods. The Java Card specification explicitly "does not include support for native methods" ([30], §3.3) outside the API.

A.VERIFICATION

All the bytecodes are verified at least once, before the loading, before the installation or before the execution, depending on the card capabilities, in order to ensure that each bytecode is valid at execution time.

5.5.2 Consumer Device

5.5.2.1 Miscellaneous

A.ACTORS

Actors of the infrastructure (CI, EUM, SM-DP+, and MNO) securely manage their own credentials and otherwise sensitive data. In particular for the overall mobile authentication mechanism defined in 3GPP TS 33.102 [22] to be secure, certain properties need to hold that are outside the scope of the eUICC. In particular, subscriber keys need to be strongly generated and securely managed. The following assumptions are therefore stated:

- The key K is randomly generated during profile preparation and is securely transported to the Authentication Centre belonging to the MNO;
- The random challenge RAND is generated with sufficient entropy in the Authentication Centre belonging to the MNO;
- The Authentication Centre belonging to the MNO generates unique sequence numbers SQN, so that each quintuplet can only be used once;
- Triplets / quintuplets are communicated securely between MNOs for roaming.

A.APPLICATIONS

The applications shall comply with the security guidelines document for the used platform (operating system). These guideline must substantially describe the application writing style and the platform security mechanisms (e.g. security domains, application firewall) that shall be used to ensure that the applications do not harm the TOE.

5.5.2.2 Device assumptions

A.TRUSTED-PATHS-LPAd

It is assumed that the interfaces ES10a, ES10b and ES10c are trusted paths between the eUICC and LPAd, when LPAd is present and active.

6 Security Objectives

6.1 Security Objectives for the TOE

6.1.1 Java Card

6.1.1.1 IDENTIFICATION

O.SID

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

6.1.1.2 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 no application can store a reference to the APDU buffer, a global byte array created by the user through makeGlobalArray method and the byte array used for 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.REALLOCATION

The TOE shall ensure that the re-allocation of a memory block for the runtime areas of the Java Card VM does not disclose any information that was previously stored in that block.

O.RESOURCES

The TOE shall control the availability of resources for the applications. See #.RESOURCES for details.

O.ARRAY_VIEWS_CONFID

The TOE shall ensure that no application can read elements of an array view not having array view security attribute ATTR_READABLE_VIEW. The TOE shall ensure that an application can only read the elements of the array view within the bounds of the array view.

O.ARRAY_VIEWS_INTEG

The TOE shall ensure that no application can write to an array view not having array view security attribute ATTR_WRITABLE_VIEW. The TOE shall ensure that an application can only write within the bounds of the array view.

6.1.1.3 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.KEY-MNGT

The TOE shall provide a means to securely manage cryptographic keys (D.APP_KEYs, D.JCS_KEYS and D.CRYPTO). This concerns the correct generation, distribution, access and destruction of cryptographic keys. See #.KEY-MNGT.

O.PIN-MNGT

The TOE shall provide a means to securely manage PIN objects. See #.PIN-MNGT for details. This concerns at least the correct authentication of the cardholder and the PIN before having access to protected operations; the observability of the comparison between presented PIN and stored PIN.

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 counter's value is as sensitive as that of the PIN.

O.TRANSACTION

The TOE must provide a means to execute a set of operations atomically. See #.TRANSACTION 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, O.PIN-MNGT, O.TRANSACTION, O.PIN-MNGT 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.

6.1.1.4 OBJECT DELETION

O.OBJ-DELETION

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

6.1.1.5 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).

6.1.1.6 Additional security objectives for the TOE

Four security objectives for the operational environment defined in the PP JCS [1] have been transformed in security objectives for the TOE:

- OE.SCP.IC
- OE.SCP.SUPPORT
- OE.SCP.RECOVERY
- OE.CARD_MANAGEMENT

O.SCP.SUPPORT

The TOE shall support the following functionalities:

- o 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.
- o It provides secure low-level cryptographic processing to the Java Card System and Global Platform.
- o It supports the needs for any update to a single persistent object or class field to be atomic, and a low-level transaction mechanism.

- 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).
- It allows the S.PPE, S.PPI, and S.TELECOM 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).
- It provides a means to perform memory operations atomically for S.PPE, S.PPI, and S.TELECOM.

O.SCP.IC

The SCP shall possess IC security features. It shall provide all IC security features against physical attacks. 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. 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.CARD_MANAGEMENT

The card manager shall control the access to card management functions such as the installation, update or deletion of applets. 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.

O.PATCH_LOADING

The TOE shall provide a secure patch code loading mechanism. The data to be loaded are encrypted. Once these data loaded, the integrity (SHA256) of the modified code is update and compared to the provided one in the patch package.

6.1.1.7 Additional objective for Sensitive Array package

O.SENSITIVE_ARRAYS_INTEG

The TOE shall provide to applet a means to securely compare two byte arrays, i.e. countermeasures against the following attacks: timing attack, comparison loop interrupted and result corrupted.

This objective ensure that no residual information is available from this operation to attackers. The operation of the comparison maintain the confidentiality of the compared arrays.

6.1.2 Consumer Device

6.1.2.1 eUICC proof of identity

O.PROOF_OF_IDENTITY

The TOE ensures that the eUICC is identified by a unique EID, based on the hardware identification of the eUICC. The eUICC must provide a cryptographic means to prove its identity to off-card actors, based on this EID.

Application Note:

This proof is obtained by including the EID value in the eUICC certificate, which is signed by the eUICC Manufacturer.

6.1.2.2 Platform services

O.OPERATE

The PPE, PPI and Telecom framework belonging to the TOE shall ensure the correct operation of their security functions.

O.API

The Platform code belonging to the TOE shall provide an API to

- provide atomic transaction to its services, and
- control the access to its services. The TOE must prevent the unauthorised use of commands.

6.1.2.3 Data protection

O.DATA-CONFIDENTIALITY

The TOE shall avoid unauthorised disclosure of the following data when stored and manipulated by the TOE:

- D.SK.EUICC.ECDSA;
- D.SECRETS;
- The secret keys which are part of the following keysets:
- D.MNO_KEYS,
- D.PROFILE_NAA_PARAMS. Application Note 11: Amongst the components of the TOE,
- PPE, PPI and Telecom Framework must protect the confidentiality of the sensitive data they process, while
- applications must use the protection mechanisms provided by the Runtime Environment. This objective includes resistance to side channel attacks.

O.DATA-INTEGRITY

The TOE shall avoid unauthorised modification of the following data when managed or manipulated by the TOE:

- The following keysets:
  - D.MNO_KEYS;
- Profile data:
  - D.PROFILE_NAA_PARAMS,
  - D.PROFILE_IDENTITY,

- D.PROFILE_POLICY_RULES,
- D.PROFILE_USER_CODES;

- o Management data:
  - D.PLATFORM_DATA,
  - D.DEVICE_INFO,
  - D.PLATFORM_RAT;
- o Identity management data:
  - D.SK.EUICC.ECDSA,
  - D.CERT.EUICC.ECDSA,
  - D.PK.CI.ECDSA,
  - D.EID,
  - D.CERT.EUM.ECDSA,
  - D.CRLs,
  - D.SECRETS.

Application Note:

Amongst the components of the TOE,

- o Platform Support Functions and Telecom Framework must protect the integrity of the sensitive data they process, while
- o applications must use the integrity protection mechanisms provided by the Runtime Environment.

6.1.2.4 Connectivity

O.ALGORITHMS

The eUICC shall provide a mechanism for the authentication to the mobile networks.

6.1.2.5 Platform support functions

O.PPE-PPI

The TOE shall provide the functionalities of platform management (loading, installation, enabling, disabling, and deletion of applications) in charge of the life-cycle of the whole eUICC and installed applications, as well as the corresponding authorization control, provided by the Profile Policy Enabler (PPE) and the Profile Package Interpreter (PPI). In particular, the PPE ensures that:

- o There is only one ISD-P enabled at a time;
- o Verification that a Profile containing PPRs is authorised by the RAT;
- o Enforcement of the PPRs of a Profile. The PPI translates the Profile Package data as defined in SIMalliance eUICC Profile Package Specification into an installed Profile using the specific internal format of the target eUICC. This functionality shall rely on the Runtime Environment secure services for package loading, application installation and deletion. Application Note 8: The PPE and PPI will in practice be tightly connected with the rest of the TOE, which in return shall very likely rely on the PPE and PPI for the effective enforcement of some of its security functions. The Platform guarantees that only the ISD-R or the Service Providers (SM-DP+, MNO) owning a Security Domain with the appropriate privilege can manage the applications on the card associated with its Security Domain. This is done accordingly with PPR and RAT. The actor performing the operation must beforehand authenticate with the Security Domain.

O.eUICC-DOMAIN-RIGHTS

The TOE shall ensure that unauthorized actors shall not get access or change personalized MNO-SD keys. Modification of this Security Domain keyset is restricted to its corresponding owner (MNO OTA Platform). In the same manner, the TOE shall ensure that only the legitimate owner of each Security Domain can access or change its confidential or integrity-sensitive data, such as for instance identity management data (for ECASD) or D.PROFILE_NAA_PARAMS (for ISD-P). This domain separation capability relies upon the Runtime Environment protection of applications.

O.SECURE-CHANNELS

The eUICC shall maintain secure channels between

- ISD-R and SM-DP+;
- MNO-SD and MNO OTA Platform. The TOE shall ensure at any time:
- that incoming messages are properly provided unaltered to the corresponding Security Domain;
- that any response messages are properly returned to the off-card entity. Communications shall be protected from unauthorized disclosure, modification and replay. This protection mechanism shall rely on the communication protection measures provided by the Runtime Environment and the PPE/PatchI (see O.PPE-PPI).

O.INTERNAL-SECURE-CHANNELS

The TOE ensures that the communication shared secrets transmitted from the ECASD to the ISD-R or ISD-P are protected from unauthorized disclosure or modification. This protection mechanism shall rely on the communication protection measures provided by the Runtime Environment.

6.2 Security Objectives for the Operational Environment

The Platform OE defined in this ST, from the PP, are integrated to the Javacard Objectives, as the plateform is in the TOE. Only OE.TRUSTED-PATHS-LPAd is kept for device interface.

6.2.1 Java Card

This section introduces the security objectives to be achieved by the environment. Four security objectives for the operational environment from the PP JCS [1] have been transformed in security objectives for the TOE:

- OE.SCP.SUPPORT
- OE.SCP.IC
- OE.SCP.RECOVERY
- OE.CARD_MANAGEMENT

OE.VERIFICATION

All the bytecodes shall be verified at least once, before the loading, before the installation or before the execution, depending on the card capabilities, in order to ensure that each bytecode is valid at execution time. See #.VERIFICATION for details. Additionally, the applet shall follow all the recommendations, if any, mandated in the platform guidance for maintaining the isolation property of the platform.

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 Security Target.

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.

OE.CAP_FILE

No CAP file loaded post-issuance shall contain native methods.

6.2.2 Consumer Device

6.2.2.1 Actors

OE.CI

The Certificate Issuer is a trusted third-party for the purpose of authentication of the entities of the system. The CI provides certificates for the EUM, SM-DS and SM-DP+. The CI must ensure the security of its own private keys.

OE.SM-DPplus

The SM-DP+ shall be a trusted actor responsible for the data preparation and the associated OTA servers. The SM-DP+ site must be accredited following GSMA SAS. It must ensure the security of the profiles it manages and loads into the eUICC, including but not limited to:

- MNO keys including OTA keys (telecom keys either generated by the SM-DP+ or by the MNO),
- Application Provider Security Domain keys (APSD keys),
- Controlling Authority Security Domain keys (CASD keys). The SM-DP+ must ensure that any key used in ISD-P are securely generated before they are transmitted to the eUICC. The SM-DP+ must ensure that any key used in ISD-P are not compromised before they are transmitted to the eUICC. The security of the ISD-P token verification keys must be ensured by a well defined security policy that covers generation, storage, distribution, destruction and recovery. This policy is enforced by the SM-DP+ in collaboration with the personalizer.

Application Note:

The SM-DP+ replaces the OE.PERSONALIZER as defined in [4].

OE.MNO

The MNOs must ensure that any key used in the profile (ISD-P, MNO SD, and any other SSD) are securely generated before they are transmitted on the eUICC via the MNO OTA Platform. The MNOs must ensure that any key used in the profile (ISD-P, MNO SD, and any other SSD) are not compromised before they are transmitted on the eUICC via the MNO OTA Platform. Administrators of the mobile operator OTA servers shall be trusted people. They shall be trained to use and administer those servers. They have the means and the equipment to perform their tasks. They must be aware of the sensitivity of the assets they manage and the responsibilities associated with the administration of OTA servers. OTA Platform communication on ES6 makes use of at least a minimum security settings defined for ES5 in [3], section 2.4.

Application Note:

One possible realisation of this assumption is the enforcement of security rules defined in an OTA server security guidance document with regular site inspections to check the applicability of the rules.

6.2.2.2 Profile

OE.MNO-SD

The Security Domain U.MNO-SD must use the secure channel SCP80/81 provided by the TOE according to [3].

OE.APPLICATIONS

The applications shall comply with the security guidelines document for the platform (operating system) used. These guideline must substantially describe the application writing style and the platform security mechanisms (e.g. security domains, application firewall) that shall be used to ensure that the applications do not harm the TOE.

6.2.2.3 Platform

OE.TRUSTED-PATHS-LPAd

The interfaces ES10a, ES10b and ES10c are trusted paths between the eUICC and LPAd, when LPAd is present and active.

6.3 Security Objectives Rationale

6.3.1 Threats

6.3.1.1 Java Card

<u>**CONFIDENTIALITY**</u>

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.

An applet might share data buffer with another applet using array views without the array view security attribute ATTR_READABLE_VIEW. The disclosure of data of the applet creating the array view is prevented by the security object O.ARRAY_VIEWS_CONFID.

Finally, any attempt to read a piece of information that was previously used by an application but has been logically deleted is countered by the O.REALLOCATION objective. That objective states that any information that was formerly stored in a memory block shall be cleared before the block is reused.

T.CONFID-JCS-CODE This threat is countered by the list of properties described in the (#.VERIFICATION) security aspect. Bytecode verification ensures that each of the instructions used on the Java Card platform is used for its intended purpose and in the intended scope of accessibility. As none of those instructions enables reading a piece of code, no Java Card applet can therefore be executed to disclose a piece of code. Native applications are also harmless because of the objective O.NATIVE, so no application can be run to disclose a piece of code.

The (#.VERIFICATION) security aspect is addressed in this ST by the objective for the environment OE.VERIFICATION.

The objectives O.CARD_MANAGEMENT and OE.VERIFICATION contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively.

T.CONFID-JCS-DATA This threat is covered by bytecode verification (OE.VERIFICATION) and the isolation commitments stated in the (O.FIREWALL) security objective. This latter objective also relies in its turn on the correct identification of applets stated in (O.SID). Moreover, as the firewall is dynamically enforced, it shall never stop operating, as stated in the (O.OPERATE) objective.

As the firewall is a software tool automating critical controls, the objective O.ALARM asks for it to provide clear warning and error messages, so that the appropriate counter-measure can be taken.

The objectives O.CARD_MANAGEMENT and OE.VERIFICATION contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively.

The objectives O.SCP.RECOVERY and O.SCP.SUPPORT are intended to support the O.OPERATE and O.ALARM objectives of the TOE, so they are indirectly related to the threats that these latter objectives contribute to counter.

INTEGRITY

T.INTEG-APPLI-CODE This threat is countered by the list of properties described in the (#.VERIFICATION) security aspect. Bytecode verification ensures that each of the instructions used on the Java Card platform is used for its intended purpose and in the intended scope of accessibility. As none of these instructions enables modifying a piece of code, no Java Card applet can therefore be executed to modify a piece of code. Native applications are also harmless because of the objective O.NATIVE, so no application can run to modify a piece of code.

The (#.VERIFICATION) security aspect is addressed in this configuration by the objective for the environment OE.VERIFICATION.

The objectives O.CARD_MANAGEMENT and OE.VERIFICATION contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively.

The objective OE.CODE-EVIDENCE contributes to cover this threat by ensuring that integrity and authenticity evidences exist for the application code loaded into the platform.

T.INTEG-APPLI-CODE.LOAD This threat is countered by the security objective O.LOAD which ensures that the loading of packages is done securely and thus preserves the integrity of packages code. The objective OE.CODE-EVIDENCE contributes to cover this threat by ensuring that the application code loaded into the platform has not been changed after code verification, which ensures code integrity and authenticity. By controlling the access to card management functions such as the installation, update or deletion of applets the objective O.CARD_MANAGEMENT contributes to cover this threat.

T.INTEG-APPLI-DATA This threat is countered by bytecode verification (OE.VERIFICATION) and the isolation commitments stated in the (O.FIREWALL) objective. This latter objective also relies in its turn on the correct identification of applets stated in (O.SID). Moreover, as the firewall is dynamically enforced, it shall never stop operating, as stated in the (O.OPERATE) objective.

As the firewall is a software tool automating critical controls, the objective O.ALARM asks for it to provide clear warning and error messages, so that the appropriate counter-measure can be taken. 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.

An applet might share data buffer with another applet using array views without the array view security attribute ATTR_WRITABLE_VIEW. The integrity of data of the applet creating the array view is ensured by the security objective O.ARRAY_VIEWS_INTEG.

Finally, any attempt to read a piece of information that was previously used by an application but has been logically deleted is countered by the O.REALLOCATION objective. That objective states that any information that was formerly stored in a memory block shall be cleared before the block is reused.

T.INTEG-APPLI-DATA.LOAD This threat is countered by the security objective O.LOAD which ensures that the loading of packages is done securely and thus preserves the integrity of applications data. The objective OE.CODE-EVIDENCE contributes to cover this threat by ensuring that the application code loaded into the platform has not been changed after code verification, which ensures code integrity and authenticity. By controlling the access to card management functions such as the installation, update or deletion of applets the objective O.CARD_MANAGEMENT contributes to cover this threat.

T.INTEG-JCS-CODE This threat is countered by the list of properties described in the (#.VERIFICATION) security aspect. Bytecode verification ensures that each of the instructions used on the Java Card platform is used for its intended purpose and in the intended scope of accessibility. As none of these instructions enables modifying a piece of code, no Java Card applet can therefore be executed to modify a piece of code. Native applications are also harmless because of the objective O.NATIVE, so no application can be run to modify a piece of code. The (#.VERIFICATION) security aspect is addressed in this configuration by the objective for the environment OE.VERIFICATION. The objectives O.CARD_MANAGEMENT and OE.VERIFICATION contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively. The objective OE.CODE-EVIDENCE contributes to cover this threat by ensuring that the application code loaded into the platform has not been changed after code verification, which ensures code integrity and authenticity.

T.INTEG-JCS-DATA This threat is countered by bytecode verification (OE.VERIFICATION) and the isolation commitments stated in the (O.FIREWALL) objective. This latter objective also relies in its turn on the correct identification of applets stated in (O.SID). Moreover, as the firewall is dynamically enforced, it shall never stop operating, as stated in the (O.OPERATE) objective. As the firewall is a software tool automating critical controls, the objective O.ALARM asks for it to provide clear warning and error messages, so that the appropriate counter-measure can be taken. The objectives O.CARD_MANAGEMENT and OE.VERIFICATION contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively. The objective OE.CODE-EVIDENCE contributes to cover this threat by ensuring that the application code loaded into the platform has not been changed after code verification, which ensures code integrity and authenticity. The objectives O.SCP.RECOVERY and O.SCP.SUPPORT are intended to support the O.OPERATE and O.ALARM objectives of the TOE, so they are indirectly related to the threats that these latter objectives contribute to counter.

IDENTITY USURPATION

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.

UNAUTHORIZED EXECUTION

T.EXE-CODE.1 Unauthorized execution of a method is prevented by the objective OE.VERIFICATION. This threat particularly concerns the point (8) of the security aspect #VERIFICATION (access modifiers and scope of accessibility for classes, fields and methods). The O.FIREWALL objective is also concerned, because it prevents the execution of non-shareable methods of a class instance by any subject apart from the class instance owner.

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.CAP_FILE also covers this threat by ensuring that no CAP files containing native code 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).

DENIAL OF SERVICE

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.

It should be noticed that, for what relates to CPU usage, the Java Card platform is single-threaded and it is possible for an ill-formed application (either native or not) to monopolize the CPU. However, a smart card can be physically interrupted (card removal or hardware reset) and most CADs implement a timeout policy that prevent them from being blocked should a card fails to answer. That point is out of scope of this Security Target, though.

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.

CARD MANAGEMENT

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.

SERVICES

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.

MISCELLANEOUS

T.PHYSICAL Covered by O.SCP.IC. Physical protections rely on the underlying platform and are therefore an environmental issue. This threat is partially covered by the security objective O.SENSITIVE_ARRAYS_INTEG which requires the TOE to detect and notify the application if any unauthorized modification of the integrity on array bytes comparison through physical attacks occurred.

Additional threats

T.PATCH_LOADING This threat is covered by O.PATCH_LOADING security objective.

If an attacker tries to avoid the loading of a patch or alter a patch (during loading or once loaded), O.PATCH_LOADING ensures trustable identification and authentication (static signature) data of the loaded patch are returned by the TOE. This information enables to check the presence of the genuine patch. Moreover, O.PATCH_LOADING, ensures authentication of the entity loading the patch before the patch is loaded in the TOE.

6.3.1.2 Consumer Device

Identity tampering

T.UNAUTHORIZED-IDENTITY-MNG O.PPE-PPI and O.eUICC-DOMAIN-RIGHTS covers this threat by providing an access control policy for ECASD content and functionality. The on-card access control policy relies upon the underlying Runtime Environment, which ensures confidentiality and integrity of application data.

T.IDENTITY-INTERCEPTION O.INTERNAL-SECURE-CHANNELS ensures the secure transmission of the shared secrets from the ECASD to ISD-R and ISD-P. These secure channels rely upon the underlying Runtime Environment, which protects the applications communications. OE.CI ensures that the CI root will manage securely its credentials off-card.

Unauthorized profile and platform management

T.UNAUTHORIZED-PROFILE-MNG This threat is covered by requiring authentication and authorization from the legitimate actors:

- O.PPE-PPI and O.eUICC-DOMAIN-RIGHTS ensure that only authorized and authenticated actors (SM-DP+ and MNO OTA Platform) will access the Security Domains functions and content;
- OE.SM-DPplus and OE.MNO protect the corresponding credentials when used off-card. The authentication is supported by corresponding secure channels:
- O.SECURE-CHANNELS and O.INTERNAL-SECURE-CHANNELS provide a secure channel for communication with SM-DP+ and a secure channel for communication with MNO OTA Platform. These secure channels rely upon the underlying Runtime Environment, which protects the applications communications. Since the MNO-SD Security Domain is not part of the TOE, the operational environment has to guarantee that it will use securely the SCP80/81

secure channel provided by the TOE (OE.MNO-SD). In order to ensure the secure operation of the Application Firewall, the following objectives for the operational environment are also required:

- compliance to security guidelines for applications (OE.VERIFICATION).

T.UNAUTHORIZED-PLATFORM-MNG This threat is covered by requiring authentication and authorization from the legitimate actors:

- O.PPE-PPI and O.eUICC-DOMAIN-RIGHTS ensure that only authorized and authenticated actors will access the Security Domains functions and content. The on-card access control policy relies upon the underlying Runtime Environment, which ensures confidentiality and integrity of application data. In order to ensure the secure operation of the Application Firewall, the following objectives for the operational environment are also required:
- compliance to security guidelines for applications (OE.VERIFICATION).

T.PROFILE-MNG-INTERCEPTION Commands and profiles are transmitted by the SM-DP+ to its on-card representative (ISD-P), while profile data (including meta-data such as PPRs) is also transmitted by the MNO OTA Platform to its on-card representative (MNO-SD). Consequently, the TSF ensures:

- Security of the transmission to the Security Domain (O.SECURE-CHANNELS and O.INTERNAL-SECURE-CHANNELS) by requiring authentication from SM-DP+ and MNO OTA Platforms, and protecting the transmission from unauthorized disclosure, modification and replay. These secure channels rely upon the underlying Runtime Environment, which protects the applications communications. Since the MNO-SD Security Domain is not part of the TOE, the operational environment has to guarantee that it will securely use the SCP80/81 secure channel provided by the TOE (OE.MNO-SD). OE.SM-DPplus and OE.MNO ensure that the credentials related to the secure channels will not be disclosed when used by off-card actors.

T.PROFILE-MNG-ELIGIBILITY Device Info and eUICCInfo2, transmitted by the eUICC to the SM-DP+, are used by the SM-DP+ to perform the Eligibility Check prior to allowing profile download onto the eUICC. Consequently, the TSF ensures:

- Security of the transmission to the Security Domain (O.SECURE-CHANNELS and O.INTERNAL-SECURE-CHANNELS) by requiring authentication from SM-DP+, and protecting the transmission from unauthorized disclosure, modification and replay. These secure channels rely upon the underlying Runtime Environment, which protects the applications communications. OE.SM-DPplus ensures that the credentials related to the secure channels will not be disclosed when used by off-card actors. O.DATA-INTEGRITY ensure that the integrity of Device Info and eUICCInfo2 is protected at the eUICC level.

eUICC cloning

T.UNAUTHORIZED-eUICC O.PROOF_OF_IDENTITY guarantees that the off-card actor can be provided with a cryptographic proof of identity based on an EID. O.PROOF_OF_IDENTITY guarantees this EID uniqueness by basing it on the eUICC hardware identification (which is unique due to O.IC.PROOF_OF_IDENTITY).

Unauthorized access to the mobile network

T.UNAUTHORIZED-MOBILE-ACCESS The objective O.ALGORITHMS ensures that a profile may only access the mobile network using a secure authentication method, which prevents impersonation by an attacker.

Second level threats

T.LOGICAL-ATTACK This threat is covered by controlling the information flow between Security Domains and the PPE, PPI, the Telecom Framework or any native/OS part of the TOE. As such it is covered:

- by the APIs of the TSF (O.API); the APIs of Telecom Framework, PPE and PPI shall ensure atomic transactions (O.SCP.SUPPORT). However these sensitive data are also processed by the PPE, PPI and the Telecom Framework, which are not protected by these mechanisms. Consequently,
- the TOE itself must ensure the correct operation of PPE, PPI and Telecom Framework (O.OPERATE), and
- PPE, PPI and Telecom Framework must protect the confidentiality and integrity of the sensitive data they process, while applications must use the protection mechanisms provided by the Runtime Environment (O.DATA-CONFIDENTIALITY, O.DATA-INTEGRITY).

The following objectives for the operational environment are also required: compliance to security guidelines for applications (OE.VERIFICATION).

LPAd impersonation

T.LPAd INTERFACE EXPLOIT OE.TRUSTED-PATHS-LPAd ensures that the interfaces ES10a, ES10b an d ES10c are trusted paths to the LPAd.

6.3.2 Organisational Security Policies

6.3.2.1 Java Card

OSP.VERIFICATION This policy is upheld by the security objective of the environment OE.VERIFICATION which guarantees that all the bytecodes shall be verified at least once, before the loading, before the installation or before the execution in order to ensure that each bytecode is valid at execution time. This policy is also upheld by the security objective of the environment OE.CODE-

EVIDENCE which ensures that evidences exist that the application code has been verified and not changed after verification.

6.3.2.2 Consumer Device

Life-cycle

OSP.LIFE-CYCLE O.PPE-PPI ensures that there is a single ISD-P enabled at a time. O.OPERATE contributes to this OSP by ensuring that the Platform security functions are always enforced.

6.3.3 Assumptions

6.3.3.1 Java Card

A.CAP_FILE This assumption is upheld by the security objective for the operational environment OE.CAP_FILE which ensures that no CAP file 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.

6.3.3.2 Consumer Device

Miscellaneous

A.ACTORS This assumption is upheld by objectives OE.CI, OE.SM-DPplus, and OE.MNO, which ensure that credentials and otherwise sensitive data will be managed correctly by each actor of the infrastructure.

A.APPLICATIONS This assumption is directly upheld by objective OE.APPLICATIONS.

Device assumptions

A.TRUSTED-PATHS-LPAd This assumption is upheld by OE.TRUSTED-PATHS-LPAd.

6.3.4 SPD and Security Objectives

Threats	Security Objectives	Rationale
T.CONFID-APPLI-DATA	OE.VERIFICATION, O.SID, O.OPERATE, O.FIREWALL, O.GLOBAL_ARRAYS_CONFID, O.ALARM, O.TRANSACTION, O.CIPHER, O.PIN-MNGT, O.KEY-MNGT, O.REALLOCATION, O.SCP.RECOVERY, O.SCP.SUPPORT, O.CARD_MANAGEMENT, O.ARRAY_VIEWS_CONFID, O.RNG	Section 6.3.1
T.CONFID-JCS-CODE	OE.VERIFICATION, O.NATIVE, O.CARD_MANAGEMENT	Section 6.3.1

T.CONFID-JCS-DATA	OE.VERIFICATION, O.SID, O.OPERATE, O.FIREWALL, O.ALARM, O.SCP.RECOVERY, O.SCP.SUPPORT, O.CARD_MANAGEMENT	Section 6.3.1
T.INTEG-APPLI-CODE	OE.VERIFICATION, O.NATIVE, OE.CODE-EVIDENCE, O.CARD_MANAGEMENT	Section 6.3.1
T.INTEG-APPLI-CODE.LOAD	O.LOAD, OE.CODE-EVIDENCE, O.CARD_MANAGEMENT	Section 6.3.1
T.INTEG-APPLI-DATA	OE.VERIFICATION, O.SID, O.OPERATE, O.FIREWALL, O.GLOBAL_ARRAYS_INTEG, O.ALARM, O.TRANSACTION, O.CIPHER, O.PIN-MNGT, O.KEY-MNGT, O.REALLOCATION, O.SCP.RECOVERY, O.SCP.SUPPORT, OE.CODE-EVIDENCE, O.CARD_MANAGEMENT, O.ARRAY_VIEWS_INTEG, O.RNG	Section 6.3.1
T.INTEG-APPLI-DATA.LOAD	O.LOAD, OE.CODE-EVIDENCE, O.CARD_MANAGEMENT	Section 6.3.1
T.INTEG-JCS-CODE	OE.VERIFICATION, O.NATIVE, OE.CODE-EVIDENCE, O.CARD_MANAGEMENT	Section 6.3.1
T.INTEG-JCS-DATA	OE.VERIFICATION, O.SID, O.OPERATE, O.FIREWALL, O.ALARM, O.SCP.RECOVERY, O.SCP.SUPPORT, OE.CODE-EVIDENCE, O.CARD_MANAGEMENT	Section 6.3.1
T.SID.1	O.FIREWALL, O.GLOBAL_ARRAYS_CONFID, O.GLOBAL_ARRAYS_INTEG, O.INSTALL, O.SID, O.CARD_MANAGEMENT	Section 6.3.1
T.SID.2	O.SID, O.OPERATE, O.FIREWALL, O.INSTALL, O.SCP.RECOVERY, O.SCP.SUPPORT	Section 6.3.1
T.EXE-CODE.1	OE.VERIFICATION, O.FIREWALL	Section 6.3.1
T.EXE-CODE.2	OE.VERIFICATION	Section 6.3.1
T.NATIVE	OE.VERIFICATION, O.NATIVE, OE.CAP_FILE	Section 6.3.1
T.RESOURCES	O.INSTALL, O.OPERATE, O.RESOURCES, O.SCP.RECOVERY, O.SCP.SUPPORT	Section 6.3.1
T.DELETION	O.DELETION, O.CARD_MANAGEMENT	Section 6.3.1
T.INSTALL	O.INSTALL, O.LOAD, O.CARD_MANAGEMENT	Section 6.3.1
T.OBJ-DELETION	O.OBJ-DELETION	Section 6.3.1
T.PHYSICAL	O.SCP.IC, O.SENSITIVE_ARRAYS_INTEG	Section 6.3.1
T.PATCH_LOADING	O.PATCH_LOADING	Section 6.3.1
T.UNAUTHORIZED-IDENTITY-MNG	O.eUICC-DOMAIN-RIGHTS, O.PPE-PPI	Section 6.3.1
T.IDENTITY-INTERCEPTION	OE.CI, O.INTERNAL-SECURE-CHANNELS	Section 6.3.1
T.UNAUTHORIZED-PROFILE-MNG	O.eUICC-DOMAIN-RIGHTS, OE.SM-DPplus, OE.MNO, O.PPE-PPI, O.SECURE-CHANNELS, O.INTERNAL-SECURE-CHANNELS, OE.MNO-SD, OE.VERIFICATION	Section 6.3.1
T.UNAUTHORIZED-PLATFORM-MNG	O.eUICC-DOMAIN-RIGHTS, O.PPE-PPI, OE.VERIFICATION	Section 6.3.1

T.PROFILE-MNG-INTERCEPTION	OE.SM-DPplus, OE.MNO, O.SECURE-CHANNELS, O.INTERNAL-SECURE-CHANNELS, OE.MNO-SD	Section 6.3.1
T.PROFILE-MNG-ELIGIBILITY	OE.SM-DPplus, O.SECURE-CHANNELS, O.INTERNAL-SECURE-CHANNELS, O.DATA-INTEGRITY	Section 6.3.1
T.UNAUTHORIZED-eUICC	O.PROOF_OF_IDENTITY	Section 6.3.1
T.UNAUTHORIZED-MOBILE-ACCESS	O.ALGORITHMS	Section 6.3.1
T.LOGICAL-ATTACK	O.DATA-CONFIDENTIALITY, O.DATA-INTEGRITY, O.API, O.OPERATE, O.SCP.SUPPORT, OE.VERIFICATION	Section 6.3.1
T.LPAd INTERFACE EXPLOIT	OE.TRUSTED-PATHS-LPAd	Section 6.3.1

Table 6 Threats and Security Objectives - Coverage

Security Objectives	Threats	Rationale
O.SID	T.CONFID-APPLI-DATA, T.CONFID-JCS-DATA, T.INTEG-APPLI-DATA, T.INTEG-JCS-DATA, T.SID.1, T.SID.2	
O.FIREWALL	T.CONFID-APPLI-DATA, T.CONFID-JCS-DATA, T.INTEG-APPLI-DATA, T.INTEG-JCS-DATA, T.SID.1, T.SID.2, T.EXE-CODE.1	
O.GLOBAL_ARRAYS_CONFID	T.CONFID-APPLI-DATA, T.SID.1	
O.GLOBAL_ARRAYS_INTEG	T.INTEG-APPLI-DATA, T.SID.1	
O.NATIVE	T.CONFID-JCS-CODE, T.INTEG-APPLI-CODE, T.INTEG-JCS-CODE, T.NATIVE	
O.REALLOCATION	T.CONFID-APPLI-DATA, T.INTEG-APPLI-DATA	
O.RESOURCES	T.RESOURCES	
O.ARRAY_VIEWS_CONFID	T.CONFID-APPLI-DATA	
O.ARRAY_VIEWS_INTEG	T.INTEG-APPLI-DATA	
O.ALARM	T.CONFID-APPLI-DATA, T.CONFID-JCS-DATA, T.INTEG-APPLI-DATA, T.INTEG-JCS-DATA	
O.CIPHER	T.CONFID-APPLI-DATA, T.INTEG-APPLI-DATA	
O.KEY-MNGT	T.CONFID-APPLI-DATA, T.INTEG-APPLI-DATA	
O.PIN-MNGT	T.CONFID-APPLI-DATA, T.INTEG-APPLI-DATA	
O.TRANSACTION	T.CONFID-APPLI-DATA, T.INTEG-APPLI-DATA	
O.RNG	T.CONFID-APPLI-DATA, T.INTEG-APPLI-DATA	
O.OBJ-DELETION	T.OBJ-DELETION	
O.DELETION	T.DELETION	
O.LOAD	T.INTEG-APPLI-CODE.LOAD, T.INTEG-APPLI-DATA.LOAD, T.INSTALL	

Security Objectives	Threats	Rationale
O.INSTALL	T.SID.1, T.SID.2, T.RESOURCES, T.INSTALL	
O.SCP.SUPPORT	T.CONFID-APPLI-DATA, T.CONFID-JCS-DATA, T.INTEG-APPLI-DATA, T.INTEG-JCS-DATA, T.SID.2, T.RESOURCES, T.LOGICAL-ATTACK	
O.SCP.IC	T.PHYSICAL	
O.SCP.RECOVERY	T.CONFID-APPLI-DATA, T.CONFID-JCS-DATA, T.INTEG-APPLI-DATA, T.INTEG-JCS-DATA, T.SID.2, T.RESOURCES	
O.CARD_MANAGEMENT	T.CONFID-APPLI-DATA, T.CONFID-JCS-CODE, T.CONFID-JCS-DATA, T.INTEG-APPLI-CODE, T.INTEG-APPLI-CODE.LOAD, T.INTEG-APPLI-DATA, T.INTEG-APPLI-DATA.LOAD, T.INTEG-JCS-CODE, T.INTEG-JCS-DATA, T.SID.1, T.DELETION, T.INSTALL	
O.PATCH_LOADING	T.PATCH_LOADING	
O.SENSITIVE_ARRAYS_INTEG	T.PHYSICAL	
O.PROOF_OF_IDENTITY	T.UNAUTHORIZED-eUICC	
O.OPERATE	T.CONFID-APPLI-DATA, T.CONFID-JCS-DATA, T.INTEG-APPLI-DATA, T.INTEG-JCS-DATA, T.SID.2, T.RESOURCES, T.LOGICAL-ATTACK	
O.API	T.LOGICAL-ATTACK	
O.DATA-CONFIDENTIALITY	T.LOGICAL-ATTACK	
O.DATA-INTEGRITY	T.PROFILE-MNG-ELIGIBILITY, T.LOGICAL-ATTACK	
O.ALGORITHMS	T.UNAUTHORIZED-MOBILE-ACCESS	
O.PPE-PPI	T.UNAUTHORIZED-IDENTITY-MNG, T.UNAUTHORIZED-PROFILE-MNG, T.UNAUTHORIZED-PLATFORM-MNG	
O.eUICC-DOMAIN-RIGHTS	T.UNAUTHORIZED-IDENTITY-MNG, T.UNAUTHORIZED-PROFILE-MNG, T.UNAUTHORIZED-PLATFORM-MNG	
O.SECURE-CHANNELS	T.UNAUTHORIZED-PROFILE-MNG, T.PROFILE-MNG-INTERCEPTION, T.PROFILE-MNG-ELIGIBILITY	
O.INTERNAL-SECURE-CHANNELS	T.IDENTITY-INTERCEPTION, T.UNAUTHORIZED-PROFILE-MNG, T.PROFILE-MNG-INTERCEPTION, T.PROFILE-MNG-ELIGIBILITY	
OE.VERIFICATION	T.CONFID-APPLI-DATA, T.CONFID-JCS-CODE, T.CONFID-JCS-DATA, T.INTEG-APPLI-CODE, T.INTEG-APPLI-DATA, T.INTEG-JCS-CODE, T.INTEG-JCS-DATA, T.EXE-CODE.1, T.EXE-CODE.2, T.NATIVE, T.UNAUTHORIZED-PROFILE-MNG, T.UNAUTHORIZED-PLATFORM-MNG, T.LOGICAL-ATTACK	

Security Objectives	Threats	Rationale
OE.CODE-EVIDENCE	T.INTEG-APPLI-CODE, T.INTEG-APPLI-CODE.LOAD, T.INTEG-APPLI-DATA, T.INTEG-APPLI-DATA.LOAD, T.INTEG-JCS-CODE, T.INTEG-JCS-DATA	
OE.CAP_FILE	T.NATIVE	
OE.CI	T.IDENTITY-INTERCEPTION	
OE.SM-DPplus	T.UNAUTHORIZED-PROFILE-MNG, T.PROFILE-MNG-INTERCEPTION, T.PROFILE-MNG-ELIGIBILITY	
OE.MNO	T.UNAUTHORIZED-PROFILE-MNG, T.PROFILE-MNG-INTERCEPTION	
OE.MNO-SD	T.UNAUTHORIZED-PROFILE-MNG, T.PROFILE-MNG-INTERCEPTION	
OE.APPLICATIONS		
OE.TRUSTED-PATHS-LPAd	T.LPAd INTERFACE EXPLOIT	

Table 7 Security Objectives and Threats - Coverage

Organisational Security Policies	Security Objectives	Rationale
OSP.VERIFICATION	OE.VERIFICATION, OE.CODE-EVIDENCE	Section 6.3.2
OSP.LIFE-CYCLE	O.PPE-PPI, O.OPERATE	Section 6.3.2

Table 8 OSPs and Security Objectives - Coverage

Security Objectives	Organisational Security Policies	Rationale
O.SID		
O.FIREWALL		
O.GLOBAL_ARRAYS_CONFID		
O.GLOBAL_ARRAYS_INTEG		
O.NATIVE		
O.REALLOCATION		
O.RESOURCES		
O.ARRAY_VIEWS_CONFID		
O.ARRAY_VIEWS_INTEG		
O.ALARM		
O.CIPHER		
O.KEY-MNGT		
O.PIN-MNGT		
O.TRANSACTION		

Security Objectives	Organisational Security Policies	Rationale
O.RNG		
O.OBJ-DELETION		
O.DELETION		
O.LOAD		
O.INSTALL		
O.SCP.SUPPORT		
O.SCP.IC		
O.SCP.RECOVERY		
O.CARD_MANAGEMENT		
O.PATCH_LOADING		
O.SENSITIVE_ARRAYS_INTEG		
O.PROOF_OF_IDENTITY		
O.OPERATE	OSP.LIFE-CYCLE	
O.API		
O.DATA-CONFIDENTIALITY		
O.DATA-INTEGRITY		
O.ALGORITHMS		
O.PPE-PPI	OSP.LIFE-CYCLE	
O.eUICC-DOMAIN-RIGHTS		
O.SECURE-CHANNELS		
O.INTERNAL-SECURE-CHANNELS		
OE.VERIFICATION	OSP.VERIFICATION	
OE.CODE-EVIDENCE	OSP.VERIFICATION	
OE.CAP_FILE		
OE.CI		
OE.SM-DPplus		
OE.MNO		
OE.MNO-SD		
OE.APPLICATIONS		
OE.TRUSTED-PATHS-LPAd		

Table 9 Security Objectives and OSPs - Coverage

Assumptions	Security Objectives for the Operational Environment	Rationale
A.CAP_FILE	OE.CAP_FILE	Section 6.3.3

A.VERIFICATION	OE.VERIFICATION, OE.CODE-EVIDENCE	Section 6.3.3
A.ACTORS	OE.CI, OE.SM-DPplus, OE.MNO	Section 6.3.3
A.APPLICATIONS	OE.APPLICATIONS	Section 6.3.3
A.TRUSTED-PATHS-LPAd	OE.TRUSTED-PATHS-LPAd	Section 6.3.3

Table 10 Assumptions and Security Objectives for the Operational Environment - Coverage

Security Objectives for the Operational Environment	Assumptions	Rationale
OE.VERIFICATION	A.VERIFICATION	
OE.CODE-EVIDENCE	A.VERIFICATION	
OE.CAP_FILE	A.CAP_FILE	
OE.CI	A.ACTORS	
OE.SM-DPplus	A.ACTORS	
OE.MNO	A.ACTORS	
OE.MNO-SD		
OE.APPLICATIONS	A.APPLICATIONS	
OE.TRUSTED-PATHS-LPAd	A.TRUSTED-PATHS-LPAd	

Table 11 Security Objectives for the Operational Environment and Assumptions - Coverage

7 Extended Requirements

7.1 Extended Families

7.1.1 Extended Family FPT_EMS - TOE Emanation

7.1.1.1 Description

The additional 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 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, radio emanation etc. This family describes the functional requirements for the limitation of intelligible emanations. The family FPT_EMS belongs to the Class FPT because it is the class for TSF protection. Other families within the Class FPT do not cover the TOE emanation.

7.1.1.2 Extended Components

Extended Component FPT_EMS.1

Description

This family defines requirements to mitigate intelligible emanations.

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.

Definition

FPT_EMS.1 TOE Emanation

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]

Dependencies: No dependencies.

7.1.2 *Extended Family FIA_API - Authentication Proof of Identity*

7.1.2.1 Description

To describe the IT security functional requirements of the TOE a sensitive family (FIA_API) of the Class FIA (Identification and authentication) is defined here. This family describes the functional requirements for the 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 10: The other families of the Class FIA describe only the authentication verification of users' identity performed by the TOE and do not describe the functionality of the user to prove their identity. The following paragraph defines the family FIA_API in the style of the Common Criteria part 2 (cf. [3], chapter 'Explicitly stated IT security requirements (APE_SRE)') from a TOE point of view.

7.1.2.2 Extended Components

Extended Component FIA_API.1

Description

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

Definition

FIA_API.1 Authentication Proof of Identity

FIA_API.1.1 The TSF shall provide a [assignment: authentication mechanismi] to prove the identity of the [assignment: authorized user or role] to an external entity.

Dependencies: No dependencies.

7.1.3 *Extended Family FCS_RNG - Random number generation*

7.1.3.1 Description

This family defines requirements for the generation random number where the random numbers are intended to be used for cryptographic purposes. The requirements address the type of the random number generator as defined in AIS 20/31 and quality of the random numbers. The classes of random number generators used in this family (DRG and PTG) are described in document [19].

7.1.3.2 Extended Components

Extended Component FCS_RNG.1

Description

A physical random number generator (RNG) produces the random number by a noise source based on physical random processes. A non-physical true RNG uses a noise source based on non-physical random processes like human interaction (key strokes, mouse movement). A deterministic RNG uses a random seed to produce a pseudorandom output. A hybrid RNG combines the principles of physical and deterministic RNGs.

FCS_RNG.1 Random number generation

FCS_RNG.1.1 The TSF shall provide a [selection: deterministic, hybrid deterministic, physical, hybrid physical] random number generator [selection: DRG.2, DRG.3, DRG.4, PTG.2, PTG.3] that implements [assignment: list of security capabilities of the selected RNG class].

FCS_RNG.1.2 The TSF shall provide random numbers that meet [assignment: a defined quality metric of the selected RNG Class].

Dependencies: No dependencies.

8 Security Requirements

8.1 Security Functional Requirements

The origin of SFRs can be found in chapter 3 of this ST. According to protection profiles and Common Criteria SFR definitions, selections and assigments filled by the ST author appear here in bold text.

8.1.1 Java Card

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

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 [1] (cf. Java Card System Protection Profile Collection [1]).
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, in those configurations that do not support on-card static or dynamic bytecode verification, the installation of a package that has not been bytecode verified, or that has been modified after bytecode verification.

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.

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:

Security attribute	Description/Value
Active Applets	The set of the active applets' AIDs. An active applet is an applet that is selected on at least one of the logical channels.
Applet Selection Status	"Selected" or "Deselected".
Applet's version number	The version number of an applet (package) indicated in the export file.
Class	Identifies the implementation class of the remote object.
Context	Package AID or "Java Card RE".
COD Context attribute	Delimits the space occupied in volatile memory by the data of the CLEAR_ON_DESELECT transient arrays of a package
COR Context attribute	Delimits the space occupied in volatile memory by the data of the CLEAR_ON_RESET transient arrays of a package
Current Frame Context	The lower and upper Boundary of the local variables area on the stack frame for a method and the lower and upper Boundary of the operand stack area on the stack frame for a method
Currently Active Context	Package AID or "Java Card RE".
Dependent package AID	Allows the retrieval of the Package AID and Applet's version number ([30], §4.5.2).
ExportedInfo Boolean	(indicates whether the remote object is exportable or not).
Identifier	The Identifier of a remote object or method is a number that uniquely identifies the remote object or method, respectively.
LC Selection Status	Multiselectable, Non-multiselectable or "None".
LifeTime	CLEAR_ON_DESELECT or PERSISTENT (*) or CLEAR_ON_RESET
Object Boundary	Delimits the space occupied by an object in the heap
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

Security attribute	Description/Value
	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.
Package Boundary	Delimits the space occupied by the code and the static fields of a package
Program Counter	Position of the next Bytecode to executed
Registered Applets	The set of AID of the applet instances registered on the card.
Resident Packages	The set of AIDs of the packages already loaded on the card.
Selected Applet Context	Package AID or "None".
Sharing	Standards, SIO, Java Card RE entry point or global array.
Stack Pointer	Position of the next free slot in the stack
Static Fields	Static fields of a package
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_DESELECT 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 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_T_ALOAD(O.JAVAOBJECT, field)	Read from an array component.
OP.ARRAY_T_ASTORE(O.JAVAOBJECT, field)	Write to 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.FLOW (O.CODE_PKG)	Any operation that modify the execution flow
OP.IMPORT_KEY	Import of the keys

Operation	Description
OP.INSTANCE_FIELD (O.JAVAOBJECT, field)	Read/Write a field of an instance of a class in the Java programming language.
OP.INVK_INTERFACE (O.JAVAOBJECT, method, arg1,...)	Invoke an interface method.
OP.INVK_VIRTUAL (O.JAVAOBJECT, method, arg1,...)	Invoke a virtual method (either on a class instance or an array object).
OP.JAVA (...)	Any access in the sense of [29], §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.LOCAL_STACK_ACCESS (...)	Any operation that read or write the local stack
OP.OPERAND_STACK_ACCESS (...)	Any operation that push or pop items on the operand stack
OP.PUT (S1,S2,I)	Transfer a piece of information I from S1 to S2.
OP.STATIC_FIELD (O.CODE_PKG, field)	Read/Write a static field of a class in the JAVA programming language
OP.THROW (O.JAVAOBJECT)	Throwing of an object (athrow, see [R7], §6.2.8.7).
OP.TYPE_ACCESS (O.JAVAOBJECT, class)	Invoke checkcast or instanceof on an object in order to access to classes (standard or shareable interfaces objects).

(*) For this operation, there is no accessed object. This rule enforces that shareable transient objects are not allowed, except some objects, such as COR. For more information refer to the Java Doc [32]. For instance, during the creation of an object, the JavaCardClass attribute's value is chosen by the creator.

8.1.1.1 CoreG_LC Security Functional Requirements

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

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_T_ALOAD,
- OP.ARRAY_T_ASTORE,
- 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	Security attributes
S.PACKAGE	LC Selection Status
S.JCVM	Active Applets, Currently Active Context
S.JCRE	Selected Applet Context
O.JAVAOBJECT	Sharing, Context, LifeTime

.

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

- **R.JAVA.1 ([R7], §6.2.8): S.PACKAGE may freely perform OP.ARRAY_ACCESS, OP.INSTANCE_FIELD, OP.INVK_VIRTUAL, OP.INVK_INTERFACE, OP.THROW or OP.TYPE_ACCESS upon any O.JAVAOBJECT whose Sharing attribute has value "JCRE entry point" or "global array".**
- **R.JAVA.2 ([R7], §6.2.8): S.PACKAGE may freely perform OP.ARRAY_ACCESS, OP.INSTANCE_FIELD, OP.INVK_VIRTUAL, OP.INVK_INTERFACE or OP.THROW upon any O.JAVAOBJECT whose Sharing attribute has value "Standard" and whose Lifetime attribute has value "PERSISTENT" only if O.JAVAOBJECT's Context attribute has the same value as the active context.**
- **R.JAVA.3 ([R7], §6.2.8.10): S.PACKAGE may perform OP.TYPE_ACCESS upon an O.JAVAOBJECT whose Sharing attribute has value "SIO" only if O.JAVAOBJECT is being cast into (checkcast) or is being verified as being an instance of (instanceof) an interface that extends the Shareable interface.**
- **R.JAVA.4 ([R7], §6.2.8.6): S.PACKAGE may perform OP.INVK_INTERFACE upon an O.JAVAOBJECT whose Sharing attribute has the value "SIO", and whose Context attribute has the value "Package AID", only if the invoked interface**

method extends the Shareable interface and one of the following conditions applies:

- **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 Active Applets.**
- **R.JAVA.5: S.PACKAGE may perform OP.CREATE only if the value of the Sharing parameter is "Standard".**

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

- **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.**
- **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.CAP_FILE performing OP.ARRAY_AASTORE of the reference of an O.JAVAOBJECT whose sharing attribute has value "global array" or "Temporary".**
- **4) S.CAP_FILE performing OP.PUTFIELD or OP.PUTSTATIC of the reference of an O.JAVAOBJECT whose sharing attribute has value "global array" or "Temporary"**
- **5) R.JAVA.7 ([29], §6.2.8.2): S.CAP_FILE performing OP.ARRAY_T_ASTORE into an array view without ATTR_WRITABLE_VIEW access attribute.**
- **6) R.JAVA.8 ([29], §6.2.8.2):S.CAP_FILE performing OP.ARRAY_T_ALOAD into an array view without ATTR_READABLE_VIEW access attribute.**

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 ([33], §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 ([29], §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).

([29], 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.

([29], §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 ([30], §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. ([29], §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:

It should be noticed that 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:

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

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

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

FDP_IFF.1.3/JCVM The TSF shall enforce the **none**.

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

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

Application Note:

The storage of temporary Java Card RE-owned objects references is runtime-enforced ([29], §6.2.8.1-3).

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

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**.

Application Note:

The semantics of the Java programming language requires for any object field and array position to be initialized with default values when the resource is allocated [33], §2.5.1.

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 should be performed in accordance with the rules given in [29], §4 and [30], §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 [29], §4 and [30], §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**.

Application Note:

The following rules are given as examples only. For instance, the last two rules are motivated by the fact that the Java Card API defines only transient arrays factory methods. Future versions may allow the creation of transient objects belonging to arbitrary classes; such evolution will naturally change the range of "secure values" for this component.

- The Context attribute of an O.JAVAOBJECT must correspond to that of an installed applet or be "Java Card RE".
- An O.JAVAOBJECT whose Sharing attribute is a Java Card RE entry point or a global array necessarily has "Java Card RE" as the value for its Context security attribute.
- Any O.JAVAOBJECT whose Sharing attribute value is not "Standard" has a PERSISTENT-LifeTime attribute's value.
- Any O.JAVAOBJECT whose LifeTime attribute value is not PERSISTENT has an array type as JavaCardClass attribute's value.

FMT_MSA.3/FIREWALL Static attribute initialisation

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

FMT_MSA.3.2/FIREWALL [Editorially Refined] The TSF shall not allow **any role** to specify alternative initial values to override the default values when an object or information is created.

Application Note:

FMT_MSA.3.1/FIREWALL

- Objects' security attributes of the access control policy are created and initialized at the creation of the object or the subject. Afterwards, these attributes are no longer mutable (FMT_MSA.1/JCRE). At the creation of an object (OP.CREATE), the newly created object, assuming that the FIREWALL access control SFP permits the operation, gets its Lifetime and Sharing attributes from the parameters of the operation; on the contrary, its Context attribute has a default value, which is its creator's Context attribute and AID respectively ([29], §6.1.3). There is one default value for the Selected Applet Context that is the default applet identifier's Context, and one default value for the Currently Active Context that is "Java Card RE".
- The knowledge of which reference corresponds to a temporary entry point object or a global array and which does not is solely available to the Java Card RE (and the Java Card virtual machine).

FMT_MSA.3.2/FIREWALL

- The intent is that none of the identified roles has privileges with regard to the default values of the security attributes. It should be noticed that creation of objects is an operation controlled by the FIREWALL access control SFP. The operation shall fail anyway if the created object would have had security attributes whose value violates FMT_MSA.2.1/FIREWALL_JCVM.

FMT_MSA.3/JCVM Static attribute initialisation

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

FMT_MSA.3.2/JCVM [Editorially Refined] The TSF shall not allow **any role** to specify alternative initial values to override the default values when an object or information is created.

FMT_SMF.1/Firewall Specification of Management Functions

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

- o **modify the Currently Active Context, the Selected Applet Context and the Active Applets**.

FMT_SMR.1/Firewall Security roles

FMT_SMR.1.1/Firewall The TSF shall maintain the roles
- o **Java Card RE (JCRE),**
- o **Java Card VM (JCVM)**.

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

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.

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**.

Application Note:

The events that provoke the de-allocation of a transient object are described in [29], §5.1.

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/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**.

Application Note:

A resource is allocated to the bArray object when a call to an applet's install() method is performed. There is no conflict with FDP_ROL.1 here because of the bounds on the rollback mechanism (FDP_ROL.1.2/FIREWALL): the scope of the rollback does not extend outside the execution of the install() method, and the de-allocation occurs precisely right after the return of it.

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)**.

Application Note:

The javacard.security & javacardx.crypto packages do provide secure interfaces to the cryptographic buffer in a transparent way. See javacard.security.KeyBuilder and Key interface of [32].

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**.

Application Note:

- The events that provoke the de-allocation of any transient object are described in [29], §5.1.
- The clearing of CLEAR_ON_DESELECT objects is not necessarily performed when the owner of the objects is deselected. In the presence of multiselectable applet instances, CLEAR_ON_DESELECT memory segments may be attached to applets that are active in different logical channels. Multiselectable applet instances within a same package must share the transient memory segment if they are concurrently active ([29], §4.2.

FDP_ROL.1/FIREWALL Basic rollback

FDP_ROL.1.1/FIREWALL The TSF shall enforce **the FIREWALL access control SFP and the JCVM information flow control SFP** to permit the rollback of the **operations OP.JAVA and OP.CREATE** on the **object O.JAVAOBJECT**.

FDP_ROL.1.2/FIREWALL The TSF shall permit operations to be rolled back within the **scope of a select(), deselect(), process(), install() or uninstall() call, notwithstanding the restrictions given in [29], §7.7, within the bounds of the Commit Capacity ([29], §7.8), and those described in [32]**.

Application Note:

Transactions are a service offered by the APIs to applets. It is also used by some APIs to guarantee the atomicity of some operation. This mechanism is either implemented in Java Card platform or relies on the transaction mechanism offered by the underlying platform. Some operations of the API are not conditionally updated, as documented in [32] (see for instance, PIN-blocking, PIN-checking, update of Transient objects).

FDP_RIP.1/GlobalArray Subset residual information protection

FDP_RIP.1.1/GlobalArray 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: **a user Global Array**.

Application Note:

An array resource is allocated when a call to the API method JCSystem.makeGlobalArray is performed. The Global Array is created as a transient JCRE Entry Point Object ensuring that reference to it cannot be retained by any application. On return from the method which called JCSystem.makeGlobalArray, the array is no longer available to any applet and is deleted and the memory in use by the array is cleared and reclaimed in the next object deletion cycle.

FCS_CKM.1/CM-SCP Cryptographic key generation

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

Cryptographic key generation algorithm	**Cryptographic key size**	**List of standards**
TDES	**112 bits or 168 bits**	**FIPS PUB 46-3 (ANSI X3.92), FIPS PUB 81**
ECKeyP	**from 192 to 521 bits**	**IEEE Std 1363a-2004 [R34]**
AES	**from 128 to 256 bits with a step of 64 bits**	**FIPS PUB 197**
GP Keys - TDES	**112 bits**	**GP 2.3**
GP Keys – AES	**128, 192, 256 bits**	**GP 2.3**

.

Application Note:

- The keys can be generated and diversified in accordance with [32] specification in classes KeyBuilder and KeyPair (at least Session key generation).
- This component is instantiated according to the version of the Java Card API applying to the security target and the implemented algorithms [32].
- This component is instantiated according to the version of the Global Platform GP 2.3 [35].
- This SFR also provide cryptographic services in low level architecture, under API.

FCS_CKM.4/CM-SCP Cryptographic key destruction

FCS_CKM.4.1/CM-SCP The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method **the keys are reset with the method clearKey()** that meets the following: **"Java Card API" specification [32]. The methods 'reset' and 'setKeyFormat' call the method key.clearKey() for clearing the value of each key**.

Application Note:

The keys are reset as specified in [32] Key class, with the method clearKey(). Any access to a cleared key for ciphering or signing shall throw an exception.

FCS_COP.1/Disp Cryptographic operation

FCS_COP.1.1/Disp The TSF shall perform **see table below** in accordance with a specified cryptographic algorithm **see table below** and cryptographic key sizes **see table below** that meet the following: **see table below:**

Cryptographic operation	Cryptographic algorithm	cryptographic key sizes	List of standards
signature, signature's verification, encryption and decryption	DES – TDES with Modes ECB, CBC, CFB, OFB or CTR and MAC algo 4 mode.	112 or 168 bits	FIPS PUB 46-3, ANSI X3.92, FIPS PUB 81, ISO/IEC 9797, Data integrity mechanism [37]
signature, signature's verification, encryption and decryption	AES with Modes CBC, CFB, CTR and CMAC	from 128 to 256 bits with a step of 64 bits	FIPS PUB 197 SP800-38B (CMAC)
signature	HMAC	64 bits up to 1016 bits Based on SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512	FIPS 198 The Keyed-Hash Message Authentication Code (HMAC)
Hash functions	SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512, SHA3 -224 to -512	SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512, SHA3 -224 to -512	Secure Hash Standard, FIPS PUB 180-3
signature, signature's verification	ECDSA	256, 384 and 521 bits	ANSI X9.62-1998
Key agreement	ECDH	256, 384 and 521 bits	IEEE P1363
Checksum	16-bit using the hardware co-processor	16 bits	ISO3309
Checksum	32-bit in software implementation	32 bits	ISO3309

.

Refinement:

TDES (IC)/IDEMIA has developed the algorithm using HW DES module/TDES encryption and decryption/Triple Data Encryption with implementation of the Data Encryption Standard, FIPS PUB 46-3, 25 Oct. 1999

SHA /IDEMIA has developed the algorithm/Hash function/SHA-1/No cryptographic key/Secure Hash Standard, Federal Information Processing Standards Publication 180-3, 2008, october

SHA /IDEMIA has developed the algorithm/Hash function/SHA-224/No cryptographic key/Secure Hash Standard, Federal Information Processing Standards Publication 180-3, 2008, october

SHA /IDEMIA has developed the algorithm/Hash function/SHA-256/No cryptographic key/Secure Hash Standard, Federal Information Processing Standards Publication 180-3, 2008, october

SHA /IDEMIA has developed the algorithm/Hash function/SHA-384/No cryptographic key/Secure Hash Standard, Federal Information Processing Standards Publication 180-3, 2008, october

SHA /IDEMIA has developed the algorithm/Hash function/SHA-512/No cryptographic key/Secure Hash Standard, Federal Information Processing Standards Publication 180-3, 2008, october

SHA /IDEMIA has developed the algorithm/Hash function/SHA-3/No cryptographic key/Secure Hash Standard, Federal Information Processing Standards Publication 202, 2015, august

KG /IDEMIA has developed the algorithm using HW PK accelerator/Key Generator//Between 1024 bits to 2048 bits/

AES/IDEMIA has developed the algorithm/Data encryption / decryption//128/192/256 bits/FIPS PUB 197, 2001, November

Application Note:

- The TOE shall provide a subset of cryptographic operations defined in [32] (see javacardx.crypto.Cipher and javacardx.security packages).
- This component is instantiated according to the version of the Java Card API applicable to the security target and the implemented algorithms ([32]).
- This SFR also provide cryptographic services in low level architecture, under API.

Card Security Management

FAU_ARP.1 Security alarms

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

- o **throw an exception,**
- o **lock the card session,**
- o **reinitialize the Java Card System and its data,**
- o **response with error code to S.CAD**

, 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,
- violation of the Firewall or JCVM SFPs,
- unavailability of resources,
- array overflow

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: **integrityControlledData**.

FDP_SDI.2.2/DATA Upon detection of a data integrity error, the TSF shall **increase counter of the Killcard file. If the maximum is reached the killcard is launched**.

Application Note:

The following data persistently stored by TOE have the user data attribute "integrityControlledData ":

- PINs (i.e. objects instance of class OwnerPin or subclass of interface PIN)
- Keys (i.e. objects instance of classes implemented the interface Key)
- SecureStores (i.e. objects instance of class SecureStore)
- Packages Java Card
- Patches

FPR_UNO.1 Unobservability

FPR_UNO.1.1 [Editorially Refined] The TSF shall ensure that **any user** is unable to observe the operation **all operations** on **D.PIN, D.APP_KEYs** by **any other user or subject**.

Application Note:

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

FPT_FLS.1/VM Failure with preservation of secure state

FPT_FLS.1.1/VM 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 ([29], §6.2.3) or after a proximity card (PICC) activation sequence ([29]). Behaviour of the TOE on power loss and reset is described in [29], §3.6 and §7.1. Behaviour of the TOE on RF signal loss is described in [29], §3.6.1.

FPT_TDC.1/VM Inter-TSF basic TSF data consistency

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

FPT_TDC.1.2/VM The TSF shall use

- **the rules defined in [30] specification,**
- **the API tokens defined in the export files of reference implementation**

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.

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 AID,**
- **Applet Selection Status ([30], §6.5)**.

Refinement:

"Individual users" stand 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 Note:

- 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/Firewall is attached to an IT security function rather than to a "user" of the CC terminology. The Java Card RE does not "identify" itself to the TOE, but it is 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: **for each loaded package is associated an unique Package AID**.

FIA_USB.1.3/AID The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: **The initially assigned Package AID is unchangeable**.

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**.

Application Note:

- The installer and the Java Card RE manage other TSF data such as the applet life cycle or CAP files, but this management is implementation specific. Objects in the Java programming language may also try to query AIDs of installed applets through the lookupAID(...) API method.
- The installer, applet deletion manager or even the card manager may be granted the right to modify the list of registered applets' AIDs in specific implementations (possibly needed for installation and deletion; see #.DELETION and #.INSTALL).

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**.

8.1.1.2 InstG Security Functional Requirements

This group consists of the SFRs related to the installation of the applets, which addresses security aspects outside the runtime. The installation of applets is a critical phase, which lies partially out of the

Boundary of the firewall, and therefore requires specific treatment. In this PP, loading a package or installing an applet modeled as importation of user data (that is, user application's data) with its security attributes (such as the parameters of the applet used in the firewall rules).

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.

FDP_ITC.2.2/Installer The TSF shall use the security attributes associated with the imported user data.

FDP_ITC.2.3/Installer The TSF shall ensure that the protocol used provides for the unambiguous association between the security attributes and the user data received.

FDP_ITC.2.4/Installer The TSF shall ensure that interpretation of the security attributes of the imported user data is as intended by the source of the user data.

FDP_ITC.2.5/Installer The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE:

CAP file loading is allowed only if, for each dependent package, its AID attribute is equal to a resident package AID attribute, the major version attribute associated to the dependent package file is equal to the major version attribute of the resident package and the minor version attribute is equal to or less than the minor version attribute associated to the resident package ([JCVM3], §4.5.2).

Application Note:

FDP_ITC.2.1/Installer:

- 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.2.3/Installer:

- The format of the CAP file is precisely defined in [30] specifications; it contains the user data (like applet's code and data) and the security attributes altogether. Therefore there is no association to be carried out elsewhere.

FDP_ITC.2.4/Installer:

- Each package contains a package Version attribute, which is a pair of major and minor version numbers ([30], §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 ([30], §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 compatible. However, package files do have "package Version Numbers" ([30]) used to indicate binary compatibility or incompatibility between successive implementations of a package, which obviously directly concern this requirement.

FDP_ITC.2.5/Installer:

- A package may depend on (import or use data from) other packages already installed. This dependency is explicitly stated in the loaded package in the form of a list of package AIDs.
- The intent of this rule is to ensure the binary compatibility of the package with those already on the card ([30], §4.4).
- The installation (the invocation of an applet's install method by the installer) is implementation dependent ([29], §11.2).
- Other rules governing the installation of an applet, that is, its registration to make it SELECTable by giving it a unique AID, are also implementation dependent (see, for example, [29], §11).

FMT_SMR.1/Installer Security roles

FMT_SMR.1.1/Installer The TSF shall maintain the roles **S.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 [29] §11.1.4**.

Application Note:

The TOE may provide additional feedback information to the card manager in case of potential security violations (see FAU_ARP.1).

FPT_RCV.3/Installer Automated recovery without undue loss

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

Refinement:

There is no maintenance mode on the TOE.

FPT_RCV.3.2/Installer For **a failure during load/installation of a package/applet and deletion of a package/applet/object**, the TSF shall ensure the return of the TOE to a secure state using automated procedures.

FPT_RCV.3.3/Installer The functions provided by the TSF to recover from failure or service discontinuity shall ensure that the secure initial state is restored without exceeding **0%** for loss of TSF data or objects under the control of the TSF.

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

Application Note:

FPT_RCV.3.1/Installer:

- This element is not within the scope of the Java Card specification, which only mandates the behaviour 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 except from [11], p298: In this maintenance mode normal operation might be impossible or severely restricted, as otherwise insecure situations might occur. Typically, only authorised users should be allowed access to this mode but the real details of who can access this mode is a function of FMT: Security management. If FMT: Security management does not put any controls on who can access this mode, then it may be acceptable to allow any user to restore the system if the TOE enters such a state. However, in practice, this is probably not desirable as the user restoring the system has an opportunity to configure the TOE in such a way as to violate the SFRs.

FPT_RCV.3.2/Installer:

- 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 [29], §11.1.5 for possible scenarios. Precise behaviour is left to implementers. This component shall include among the listed failures the deletion of a package/applet. See ([29], 11.3.4) for possible scenarios. Precise behaviour 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 [38]) and, from the TOE's side, by events "that clear transient objects" and transactional features. See FPT_FLS.1/VM, FDP_RIP.1/TRANSIENT, FDP_RIP.1/ABORT and FDP_ROL.1/FIREWALL.

FPT_RCV.3.3/Installer:

- The quantification is implementation dependent, but some facts can be recalled here. First, the SCP ensures the atomicity of updates for fields and objects, and a power-failure during a transaction or the normal runtime does not create the loss of otherwise-permanent data, in the sense that memory on a smart card is essentially persistent with this respect (Flash). 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.

8.1.1.3 ADELG Security Functional Requirements

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

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 hold:**

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

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

- **R.JAVA.14 ([29], §11.3.4.1, Applet Instance Deletion): S.ADEL may perform OP.DELETE_APPLET upon an O.APPLET only if,**
  - **(1) S.ADEL is currently selected,**
  - **(2) there is no instance in the context of O.APPLET that is active in any logical channel and**
  - **(3) there is no O.JAVAOBJECT owned by O.APPLET such that either O.JAVAOBJECT is reachable from an applet instance distinct from O.APPLET, or O.JAVAOBJECT is reachable from a package P, or ([29], §8.5) O.JAVAOBJECT is remote reachable.**
- **R.JAVA.15 ([29], §11.3.4.1, Multiple Applet Instance Deletion): S.ADEL may perform OP.DELETE_APPLET upon several O.APPLET only if,**
  - **(1) S.ADEL is currently selected,**
  - **(2) there is no instance of any of the O.APPLET being deleted that is active in any logical channel and**
  - **(3) there is no O.JAVAOBJECT owned by any of the O.APPLET being deleted such that either O.JAVAOBJECT is reachable from an applet instance distinct from any of those O.APPLET, or O.JAVAOBJECT is reachable from a package P, or ([29], §8.5) O.JAVAOBJECT is remote reachable.**

- **R.JAVA.16 ([29], §11.3.4.2, Applet/Library Package Deletion): S.ADEL may perform OP.DELETE_PCKG upon an O.CODE_PKG only if,**
  - **(1) S.ADEL is currently selected,**
  - **(2) no reachable O.JAVAOBJECT, from a package distinct from O.CODE_PKG that is an instance of a class that belongs to O.CODE_PKG, exists on the card and**
  - **(3) there is no resident package on the card that depends on O.CODE_PKG.**
- **R.JAVA.17 ([29], §11.3.4.3, Applet Package and Contained Instances Deletion): S.ADEL may perform OP.DELETE_PCKG_APPLET upon an O.CODE_PKG only if,**
  - **(1) S.ADEL is currently selected,**
  - **(2) no reachable O.JAVAOBJECT, from a package distinct from O.CODE_PKG, which is an instance of a class that belongs to O.CODE_PKG exists on the card,**
  - **(3) there is no package loaded on the card that depends on O.CODE_PKG, and**
  - **(4) for every O.APPLET of those being deleted it holds that: (i) there is no instance in the context of O.APPLET that is active in any logical channel and (ii) there is no O.JAVAOBJECT owned by O.APPLET such that either O.JAVAOBJECT is reachable from an applet instance not being deleted, or O.JAVAOBJECT is reachable from a package not being deleted, or ([29], §8.5) O.JAVAOBJECT is remote reachable**.

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

FDP_ACF.1.4/ADEL 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 Note:

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:

Deleted freed resources (both code and data) may be reused, depending on the way they were deleted (logically or physically). Requirements on de-allocation during applet/package deletion are described in [29], §11.3.4.1, §11.3.4.2 and §11.3.4.3.

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**.

FMT_MSA.3/ADEL Static attribute initialisation

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

FMT_MSA.3.2/ADEL The TSF shall allow the **following role(s): none** to specify alternative initial values to override the default values when an object or information is created.

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**.

Application Note:

The modification of the Active Applets security attribute should be performed in accordance with the rules given in [29], §4.

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 [29], §11.3.4**.

Application Note:

- The TOE may provide additional feedback information to the card manager in case of a potential security violation (see FAU_ARP.1).
- The Package/applet instance deletion must be atomic. The "secure state" referred to in the requirement must comply with Java Card specification ([29], §11.3.4.)

8.1.1.4 ODELG Security Functional Requirements

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

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()**.

Application Note:

- Freed data resources resulting from the invocation of the method javacard.framework.JCSystem.requestObjectDeletion() may be reused. Requirements on de-allocation after the invocation of the method are described in [32].
- There is no conflict with FDP_ROL.1 here because of the bounds on the rollback mechanism: the execution of requestObjectDeletion() is not in the scope of the rollback because it must be performed in between APDU command processing, and therefore no transaction can be in progress.

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**.

Application Note:

The TOE may provide additional feedback information to the card manager in case of potential security violation (see FAU_ARP.1).

8.1.1.5 CarG Security Functional Requirements

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

Miscellaneous

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.

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

FCO_NRO.2.3/CM The TSF shall provide a capability to verify the evidence of origin of information to **recipient** given **immediate verification**.

Application Note:

FCO_NRO.2.1/CM:

- 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.2.3/CM:

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

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 an attacker. Moreover, an 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: **Load File, Dap**.

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: **the rules describing the communication protocol used by the CAD and the card for transmitting a new package, see chapter 9.3.9 [39]**.

FDP_IFF.1.3/CM The TSF shall enforce the **none**.

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

FDP_IFF.1.5/CM The TSF shall explicitly deny an information flow based on the following rules: **the rules describing the communication protocol used by the CAD and the card for transmitting a new package, see chapter 9.3.9 [39]**.

Application Note:

FDP_IFF.1.1/CM:

- The security attributes used to enforce the PACKAGE LOADING SFP are implementation dependent. More precisely, they depend on the communication protocol enforced between the CAD and the card. For instance, some of the attributes that can be used are: (1) the keys used by the subjects to encrypt/decrypt their messages; (2) the number of pieces the application package has been split into in order to be sent to the card; (3) the ordinal of each piece in the decomposition of the package, etc. See for example Appendix D of [40].

FDP_IFF.1.2/CM:

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

FDP_UIT.1/CM Data exchange integrity

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

FDP_UIT.1.2/CM [Editorially Refined] The TSF shall be able to determine on receipt of user data, whether **modification, deletion, insertion and 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 **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.

Application Note:

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

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 **AS.KEYSET_VERSION, AS.KEYSET_VALUE, Default SELECTED Privileges, AS.CMLIFECYC** to **R.Card_Manager**.

FMT_MSA.3/CM Static attribute initialisation

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 the **Card manager** 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: **Modify the following security attributes: AS.KEYSET_VERSION, AS.KEYSET_VALUE, Default SELECTED Privileges, AS.CMLIFECYC**.

FMT_SMR.1/CM Security roles

FMT_SMR.1.1/CM The TSF shall maintain the roles **Card manager**.

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 [Editorially 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:

New packages can be installed on the card only on demand of the card issuer.

Additional Security Functional Requirements for CM

FPT_TST.1 TSF testing

FPT_TST.1.1 The TSF shall run a suite of self tests **during initial start-up** 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**.

Application Note:
Namely, "stored TSF executable code" encompasses the patch and java packages. During startup, the TOE checks the integrity of the patch/java packages. To do so, the related bits should have been set accordingly.

Other self-tests are BIST (BUILT-IN SELF TESTING) mechanism, an auto test command available to check the integrity of the Secure OS and the chip after mounting on the device. The test procedure will be broken down into four subtests:

- Flash Integrity
- RAM Integrity
- Unitary Flash Read/Write

- Crypto processors integrity

FCO_NRO.2/CM_DAP Enforced proof of origin

FCO_NRO.2.1/CM_DAP The TSF shall enforce the generation of evidence of origin for transmitted **Load file** at all times.

FCO_NRO.2.2/CM_DAP The TSF shall be able to relate the **AS.KEYSET_VALUE** of the originator of the information, and the **CAP file components** of the information to which the evidence applies.

FCO_NRO.2.3/CM_DAP The TSF shall provide a capability to verify the evidence of origin of information to **recipient** given **during CAP file loading**.

Application Note:
This feature included in this st allows an Application Provider to require that their Application code to be loaded on the card shall be checked for integrity and authenticity. The DAP Verification Key is identified by the Key Version Number '73' and the Key Identifier '01'.

See description in §9.2.1 of GlobalPlatform Card Specification for more details [11].

In this implementation, DAPs are generated and verified according the one of the following schemes:

- The AES scheme specified in appendix B.2 of [8] is supported. For this scheme, the DAP Verification Key shall be a 128-bits AES key.
- The DES scheme specified in appendix B.1 of [8] is supported. For this scheme, the DAP Verification Key shall be a 112-bits DES key.

FIA_UAU.1/CM Timing of authentication

FIA_UAU.1.1/CM The TSF shall allow **Get Data, Initialize Update, Select** on behalf of the user to be performed before the user is authenticated.

FIA_UAU.1.2/CM 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/CardIssuer Single-use authentication mechanisms

FIA_UAU.4.1/CardIssuer The TSF shall prevent reuse of authentication data related to **Authentication Mechanism based on Triple-DES and/or AES**.

Application Note:
The authentication mechanism, used to open a secure channel communication with the card issuer, use a challenge freshly and randomly generated by the TOE in order to prevent reuse of a response generated by a terminal in a successful authentication attempt.

FPT_TDC.1/CM Inter-TSF basic TSF data consistency

FPT_TDC.1.1/CM The TSF shall provide the capability to consistently interpret **AS.KEYSET_VALUE** when shared between the TSF and another trusted IT product.

FPT_TDC.1.2/CM The TSF shall use **the rules defined in the GP [39] section 11.8** when interpreting the TSF data from another trusted IT product.

FCS_COP.1/CM-SCP Cryptographic operation

FCS_COP.1.1/CM-SCP The TSF shall perform **see table below** in accordance with a specified cryptographic algorithm **see table below** and cryptographic key sizes **see table below** that meet the following:

Cryptographic operation	Algorithm	Key length	Standard
TOE authentication key ISK/KMC	SCP02	112 bits	GP 2.3
TOE authentication key ISK/KMC	SCP03	128/192/256 bits	GP 2.3
SCP02 - signature, verification of signature, encryption and decryption (KEK (key encryption key) for sensitive objects such as PIN, keys … is mandatory)	TDES CBC, ECB	112 bits	SCP02 – GP 2.3
SCP02 - MAC verification or generation	MAC DES or TDES	112 bits, 168 bits	FIPS PUB 46-3, FIPS PUB 81, ISO/IEC 9797-1, PKCS#51
SCP03 - signature, verification of signature, encryption and decryption	AES	128/192/256 bits	SCP03 – GP 2.3
SCP03 - MAC verification or generation	CMAC AES	128/192/256 bits	NIST 800 38 B
SCP03 - Key derivation	CMAC AES based on KDF	128/192/256 bits	NIST 800 38 B
SCP03 - hash function	SHA-256, SHA-384, SHA-512	-	ISO 10118 3, FIPS 180 4
SCP080 - Secure communication channel with OTA Server	TDES	112 bits	TS 102 225 [12]
SCP080 - Secure communication channel with OTA Server	AES	128/192/256 bits	TS 102 226 [12]

Cryptographic operation	Algorithm	Key length	Standard
SCP81 - Secure communication channel with the Remote Administration Server	**TLS_PSK_WITH_AES_128_CBC_SHA256, TLS_PSK_WITH_NULL_SHA256**	**-**	**[13] section 3.3.2**
SCP-SGP22 Secure communication channel with the SM-DP+ for mutual authentication	**ECKA-EG**	**NIST P-256, brainpoolP256r1**	**SGP.22 FIPS PUB 186-3 Digital Signature Standard, BSI TR-03111 Version 1.11 RFC 5639**
SCP-SGP22: SCP03t Secure communication channel with the SM-DP+ for profile download	**AES**	**128 bits**	**SGP.02 [3]**

.

Additional Security Functional Requirements for patch

FDP_ACC.2/Patch Complete access control

FDP_ACC.2.1/Patch [Editorially Refined] The TSF shall enforce the **Patch Loading Access Control** on **S.TOE and for all objects** and all operations among subjects and objects covered by the SFP.

FDP_ACC.2.2/Patch 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:
This SFR enforces the access control for the patch loading and the ISK loading.

FDP_ACF.1/Patch Security attribute based access control

FDP_ACF.1.1/Patch The TSF shall enforce the **Access Control on See below** to objects based on the following:

Subject	Attribute
S.INSTALLER	**SCP opened**

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

- **S.INSTALLER is allowed to load a patch if:**
  - **SCP opened to protect D.SENSITIVE_DATA and patches.**
  - **correctly encrypted with the JSK key**

- **the memory area to be modified is genuine**.

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

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

Application Note:
The dedicated key will be JSK in case of patch loading. The patch loading uses JSK for encryption and compute an additional MAC of the patch.

The integrity of the memory area to be modified is verified, signature SHA256 before and after patch is loaded.

FDP_UCT.1/Patch Basic data exchange confidentiality

FDP_UCT.1.1/Patch The TSF shall enforce the **Patch loading access control** to **receive** user data in a manner protected from unauthorised disclosure.

Application Note:
For the Patch loading access control, the JSK is used to cipher the data transmitted.

FDP_ITC.1/Patch Import of user data without security attributes

FDP_ITC.1.1/Patch The TSF shall enforce the **Patch loading access control** when importing user data, controlled under the SFP, from outside of the TOE.

FDP_ITC.1.2/Patch The TSF shall ignore any security attributes associated with the user data when imported from outside the TOE.

FDP_ITC.1.3/Patch The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: **none.**

FCS_COP.1/Patch Cryptographic operation

FCS_COP.1.1/Patch The TSF shall perform **see table below** in accordance with a specified cryptographic algorithm **see table below** and cryptographic key sizes **see table below** that meet the following:

Cryptographic operation	Algorithm	Key length	Standard
Decryption of patch ciphered with diversified JSK	**AES mode CBC**	**128 bits**	**FIPS PUB 197**

FDP_UIT.1/Patch Data exchange integrity

FDP_UIT.1.1/Patch The TSF shall enforce the **Patch access control SFP** to **receive** user data in a manner protected from **modification** errors.

FDP_UIT.1.2/Patch [Editorially Refined] The TSF shall be able to determine on receipt of user data, whether **modification of some of the pieces of the application or runtime environment sent by the TOE developer or patch developer** 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 patch to be installed on the card to be different from the one sent by the TOE Developer.

FAU_STG.2/Patch Guarantees of audit data availability

FAU_STG.2.1/Patch The TSF shall protect the stored audit records in the audit trail from unauthorised deletion.

FAU_STG.2.2/Patch The TSF shall be able to **prevent** unauthorised modifications to the stored audit records in the audit trail.

FAU_STG.2.3/Patch The TSF shall ensure that **Patch code identification** stored audit records will be maintained when the following conditions occur: **audit storage exhaustion, failure and attack**

Application Note:
Patch loading steps are:

1. Loading patch containing the code in EEPROM (install and load commands)
2. Decryption with JSK key
3. Check the signature (SHA256) of the area to be patched if signature is OK
4. Write the new code (patch itself)
5. Computation of the new SHA signature
6. Deletion of the patch stored temporary in EEPROM

Information on the Patch code (unique identifier) is directly retrieved by GET DATA command.

Additional Security Functional Requirements for SmartCard Platform

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 function either completes successfully, or for the indicated failure scenarios, recovers to a consistent and secure state.

Additional Security Functional Requirements for the applets

FCS_RNG.1 Random number generation

FCS_RNG.1.1 The TSF shall provide a **physical** random number generator **PTG.2** that implements

- **(PTG.2.1) A total failure test detects a total failure of entropy source immediately when the RNG has started. When a total failure is detected, no random numbers will be output.**
- **(PTG.2.2) If a total failure of the entropy source occurs while the RNG is being operated, the RNG prevents the output of any internal random number that depends on some raw random numbers that have been generated after the total failure of the entropy source.**
- **(PTG.2.3) The online test shall detect non-tolerable statistical defects of the raw random number sequence (i) immediately when the RNG has started, and (ii) while the RNG is being operated. The TSF must not output any random numbers before the power-up online test has finished successfully or when a defect has been detected.**
- **(PTG.2.4) The online test procedure shall be effective to detect non-tolerable weaknesses of the random numbers soon.**
- **(PTG.2.5) The online test procedure checks the quality of the raw random number sequence. It is triggered externally. The online test is suitable for detecting non-tolerable statistical defects of the statistical properties of the raw random numbers within an acceptable period of time**.

FCS_RNG.1.2 The TSF shall provide random numbers that meet

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

<u>Additional Security Functional Requirements for Runtime Verification</u>

Stack Control

FDP_ACC.2/RV_Stack Complete access control

FDP_ACC.2.1/RV_Stack The TSF shall enforce the **Stack Access Control SFP** on **S.STACK** and all operations among subjects and objects covered by the SFP.

Refinement:

The operations involved in the policy are:

- OP.OPERAND_STACK_ACCESS
- OP.LOCAL_STACK_ACCESS

FDP_ACC.2.2/RV_Stack 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/RV_Stack Security attribute based access control

FDP_ACF.1.1/RV_Stack The TSF shall enforce the **Stack Access Control** to objects based on the following:

Subject/Object	Security attributes
S.APPLET	Active Applets, Applet Selection Status
S.STACK	Stack Pointer
S.JCVM	Current Frame Context

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

- **An Active Applet selected may freely perform OP.LOCAL_STACK_ACCESS upon stack pointer only if the index of the local variable accessed matches the Current Frame Context attribute.**
- **An Active Applet selected may freely perform OP.OPERAND_STACK_ACCESS upon Stack Pointer only if the attribute Stack Pointer matches the attribute Current Frame Context of S.JCVM**.

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

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

Application Note:
Any bytecode accessing a local variable has an index in parameter (byte or short). The first rule aims at verifying that this index is always positive and inferior to the numbers of local variables defined for this stack frame. Then the local variable slot is accessed using the index that is relative to the base of local variables for this stack frame.

Any bytecode accessing the operand stack for push or pop operations is under the control of rule 2. The second rule aims at verifying that the stack pointer is always in the range defined by the base-of-stack and top-of-stack values defined for this stack frame.

The frame context attribute is made of the following elements:

- number-of-local variables and base-of-local-variable
- base-of-stack and top-of-stack

The policies defined in this SFR are enforced dynamically, each time an operation is performed. Nevertheless, those verifications may be redundant with the ones made statically by the off-card verifier, during the applet verification stage.

FMT_MSA.1/RV_Stack Management of security attributes

FMT_MSA.1.1/RV_Stack The TSF shall enforce the **Stack Access Control SFP** to restrict the ability to **modify** the security attributes **Current Frame Context and Stack Pointer** to **the Java Card VM (S.JCVM)**.

FMT_MSA.2/RV_Stack Secure security attributes

FMT_MSA.2.1/RV_Stack The TSF shall ensure that only secure values are accepted for **Current Frame Context and Stack Pointer**.

FMT_MSA.3/RV_Stack Static attribute initialisation

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

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

FMT_SMF.1/RV_Stack Specification of Management Functions

FMT_SMF.1.1/RV_Stack The TSF shall be capable of performing the following management functions: **Modify the Current Frame Context and modify the Stack Pointer**.

Application Note:

The frame context attribute is modified on method invocation. In that case, the previous context attribute is saved on the stack. It will be restored on return of the invoked method.

Heap Access

FDP_ACC.2/RV_Heap Complete access control

FDP_ACC.2.1/RV_Heap The TSF shall enforce the **Heap Access Control SFP** on **O.CODE_PKG, O.JAVAOBJECT, S.JCVM, S.APPLET** and all operations among subjects and objects covered by the SFP.

Refinement:

The operations involved in the policy are:

- OP.ARRAY_ACCESS
- OP.INSTANCE_FIELD
- OP.STATIC_FIELD
- OP.FLOW

FDP_ACC.2.2/RV_Heap 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/RV_Heap Security attribute based access control

FDP_ACF.1.1/RV_Heap The TSF shall enforce the **Heap Access Control SFP** to objects based on the following:

Subject/Object	**Security attributes**
O.CODE_PKG	**Package Boundary**
O.JAVAOBJECT	**Object Boundary**
S.JCVM	**Program Counter**
S.APPLET	**Active Applets, Applet Selection Status**

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

- **S.APPLET may freely perform OP.ARRAY_ACCESS and OP.INSTANCE_FIELD upon any O.JAVAOBJECT if the array cell index or the instance field index match the object boundary attribute of O.JAVAOBJECT**
- **S.APPLET may freely perform OP.STATIC_FIELD upon any O.CODE_PKG if the static field index matches the Package Boundary attribute of O.CODE_PKG.**
- **S.APPLET may freely perform OP.FLOW upon O.CODE_PKG if the Program Counter attribute of S.JCVM matches the Package Boundary attribute of O.CODE_PKG**.

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

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

Application Note:
The upper and lower boundaries of any object allocated on the heap are registered (Object Boundary Attribute). Each time an object is accessed, the first rule verifies that the accessed NVM location is comprised between those two boundaries.

The second rule aims at verifying that when a static field is accessed, the index of this field is positive and inferior to the number of static fields of this package (part of Package Boundary attribute).

The third rule aims at verifying that when a change of execution flow occurs, the computed value for the newly computed value for the Program Counter is comprised within the boundaries defined for this package (part of Package Boundary Attribute). This rule does not concern invocation bytecode.

The policies defined in this SFR are enforced dynamically, each time an operation is performed. Nevertheless, those verifications may be redundant with the ones made statically by the off-card verifier, during the applet verification stage.

FMT_MSA.1/RV_Heap Management of security attributes

FMT_MSA.1.1/RV_Heap The TSF shall enforce the **Heap Access Control SFP** to restrict the ability to **modify** the security attributes **Package Boundary, Object Boundary and Program Counter** to **S.JCVM**.

FMT_MSA.2/RV_Heap Secure security attributes

FMT_MSA.2.1/RV_Heap The TSF shall ensure that only secure values are accepted for **Package Boundary, Object Boundary and Program Counter**.

FMT_MSA.3/RV_Heap Static attribute initialisation

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

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

FMT_SMF.1/RV_Heap Specification of Management Functions

FMT_SMF.1.1/RV_Heap The TSF shall be capable of performing the following management functions: **to modify the Program Counter attribute**.

Transient Control

FDP_ACC.2/RV_Transient Complete access control

FDP_ACC.2.1/RV_Transient The TSF shall enforce the **Transient Access Control SFP** on **S.APPLET, S.JCVM and O.JAVAOBJECT** and all operations among subjects and objects covered by the SFP.

Refinement:

The operation involved in the policy is:

- OP.ARRAY_ACCESS

FDP_ACC.2.2/RV_Transient 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/RV_Transient Security attribute based access control

FDP_ACF.1.1/RV_Transient The TSF shall enforce the **Transient Access Control SFP** to objects based on the following:

Subject/Object	Security Attributes
S.APPLET	Active Applets, Applet Selection Status
S.JCVM	COR Context, COD Context
O.JAVAOBJECT	LifeTime

.

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

- **S.APPLET may freely perform OP.ARRAY_ACCESS on O.JAVAOBJECT whose LifeTime attribute has value "CLEAR_ON_RESET" only if the targeted volatile memory space matches the COR Context attribute of S.JCVM**
- **S.APPLET may freely perform OP.ARRAY_ACCESS on O.JAVAOBJECT whose LifeTime attribute has value "CLEAR_ON_DESELECT" only if the targeted volatile memory space matches the COD Context attribute of S.JCVM**.

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

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

Application Note:
Each time an applet accesses a Clear On Reset (resp. Clear On Deselect) transient, these rules verify that the accessed RAM area is in the range of the Clear On Reset transients space (resp. Clear On Deselect) allocated for all the transients created by the applets of this package.

The COR context attribute represents the lower and upper limits for the Clear On Reset transient space of the active applet package. The COD context attribute represents the lower and upper limits for the Clear On Deselect transient space of the currently selected applet package.

The policies defined in this SFR are enforced dynamically, each time an operation is performed. Nevertheless, those verifications may be redundant with the ones made statically by the off-card verifier, during the applet verification stage.

FMT_MSA.1/RV_Transient Management of security attributes

FMT_MSA.1.1/RV_Transient The TSF shall enforce the **Transient Access Control SFP** to restrict the ability to **modify** the security attributes **the security attributes COR Context and COD Context** to **Java Card VM (S.JCVM)**.

FMT_MSA.2/RV_Transient Secure security attributes

FMT_MSA.2.1/RV_Transient The TSF shall ensure that only secure values are accepted for **COR Context and COD Context Security attributes of the Transient Access Control SFP**.

FMT_MSA.3/RV_Transient Static attribute initialisation

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

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

FMT_SMF.1/RV_Transient Specification of Management Functions

FMT_SMF.1.1/RV_Transient The TSF shall be capable of performing the following management functions: **modify the COR Context and COD Context Security Attributes**.

8.1.1.6 Additional Security Functional Requirement for Sensitive Array package

FDP_SDI.2/ARRAY Stored data integrity monitoring and action

FDP_SDI.2.1/ARRAY The TSF shall monitor user data stored in containers controlled by the TSF for **integrity errors** on all objects, based on the following attributes: **user data stored in arrays created by the makeIntegritySensitiveArray() method of the javacard.framework.SensitiveArrays class**.

FDP_SDI.2.2/ARRAY Upon detection of a data integrity error, the TSF shall **throw an exception**.

Application Note:
This requirement applies in particular to the arrays created by the makeIntegritySensitiveArray() method of the javacard.framework.SensitiveArrays class

8.1.2 Consumer Device

8.1.2.1 Identification and authentication

FIA_UAU.1/EXT Timing of authentication

FIA_UAU.1.1/EXT The TSF shall allow

- **application selection**
- **requesting data that identifies the eUICC**

- **user identification**
- **no additional TSF mediated actions**

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

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

Application Note:
This SFR is related to the authentication of the following external (remote) users of the TOE:

- U.SM-DPplus,
- U.MNO-OTA.

FIA_USB.1/EXT User-subject binding

FIA_USB.1.1/EXT The TSF shall associate the following user security attributes with subjects acting on the behalf of that user:

- **SM-DP+ OID is associated to S.ISD-R, acting on behalf of U.SM-DPplus**
- **MNO OID is associated to U.MNO-SD, acting on behalf of U.MNO-OTA**.

FIA_USB.1.2/EXT The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users:

- **Initial association of SM-DP+ OID and MNO OID requires U.SM-DPplus to be authenticated via "CERT.DPauth.ECDSA"**.

FIA_USB.1.3/EXT The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users:

- **change of SM-DP+ OID requires U.SM-DPplus to be authenticated via "CERT.DPauth.ECDSA"**
- **change of MNO OID is not allowed**.

Application Note:
This SFR is related to the binding of external (remote) users to local subjects or users of the TOE:

- U.SM-DP+ binds to a subject (S.ISD-R)
- U.MNO-OTA binds to an on-card user (U.MNO-SD). Here U.MNO-SD is not a subject of the TOE, but an external on-card user acting on behalf of U.MNO-OTA, which is an external off-card user. This SFR is related to the following commands:
- Initial association of the D.MNO_KEYS keyset is performed by the ES8+.ConfigureISDP command.

FIA_UAU.4/EXT Single-use authentication mechanisms

FIA_UAU.4.1/EXT The TSF shall prevent reuse of authentication data related to **the authentication mechanism used to open a secure communication channel between the eUICC and**

- **U.SM-DPplus**
- **U.MNO-OTA**.

Application Note:
This SFR is related to the authentication of external (remote) users of the TOE:

- U.SM-DPplus,
- U.MNO-OTA.

FIA_UID.1/MNO-SD Timing of identification

FIA_UID.1.1/MNO-SD The TSF shall allow **MNO-SD shall: a) Be associated to itself and remain linked to the ISD-P; b) Contain the Operator OTA Keys; c) Provide a secure OTA channel (SCP80 or SCP81 as defined in [12] and [13]); d) Have the capability to host Supplementary Security Domains** on behalf of the user to be performed before the user is identified.

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

Application Note:
This SFR is related to the identification of the local user U.MNO-SD only. It should be noted that the U.MNO-SD is identified but not authenticated. However, U.MNO-SD is installed on the TOE by the U.SM-DPplus via the subject S.ISD-R (see FDP_ACF.1/ISDR), and the binding between U.SM-DPplus and S.ISD-R requires authentication of U.SM-DP+, as described in FIA_USB.1/EXT.

FIA_USB.1/MNO-SD User-subject binding

FIA_USB.1.1/MNO-SD The TSF shall associate the following user security attributes with subjects acting on the behalf of that user: **The U.MNO-SD AID is associated to the S.ISD-P acting on behalf of U.MNO-SD**.

FIA_USB.1.2/MNO-SD The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users: **Initial association of AID requires U.SM-DP+ to be authenticated via CERT.DPauth.ECDSA**.

FIA_USB.1.3/MNO-SD The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: **no change of AID is allowed**.

Application Note:
This SFR is related to the identification of the local user U.MNO-SD. Being a local but external user of the TOE, the U.MNO-SD is bound to the S.ISD-R which is responsible for its installation during the "Profile download and install". This profile installation is controlled by the FDP_ACC.1/ISDR SFP. Being performed by the S.ISD-R, it requires authentication of the U.SM-DPplus. In order to perform operations such as PPR update, U.MNO-OTA authenticates, then sends a command to U.MNO-SD, which transmits it to S.ISD-P; the operation is eventually executed by the S.ISD-P according to the FDP_ACC.1/ISDP SFP. The identification does not depend on direct authentication of the MNO OTA Platform, but on the authentication of the S.ISD-R: The S.ISD-R installs a profile which includes a U.MNO-SD and associated keyset.

FIA_API.1 Authentication Proof of Identity

FIA_API.1.1 The TSF shall provide a **cryptographic authentication mechanism based on the EID of the eUICC** to prove the identity of the **TOE** to an external entity.

Application Note:
This proof is obtained by including the EID value in the eUICC certificate, which is signed by the eUICC Manufacturer.

FIA_UID.1/EXT Timing of identification

FIA_UID.1.1/EXT The TSF shall allow

- **application selection**
- **requesting data that identifies the eUICC**
- **none additional action**

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

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

FIA_ATD.1/User User attribute definition

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

- **CERT.DPauth.ECDSA, CERT.DPpb.ECDSA, and SM-DP+ OID belonging to U.SM-DPplus;**
- **MNO OID belonging to U.MNO-OTA;**
- **AID belonging to U.MNO-SD**.

8.1.2.2 Communication

FDP_IFC.1/SCP Subset information flow control

FDP_IFC.1.1/SCP The TSF shall enforce the **Secure Channel Protocol information flow control SFP** on

- **users/subjects:**
  - **U.SM-DPplus and S.ISD-R**
  - **U.MNO-OTA and U.MNO-SD**
- **information: transmission of commands**.

FDP_IFF.1/SCP Simple security attributes

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

- o **users/subjects:**
  - **U.SM-DPplus and S.ISD-R, with security attribute D.SECRETS**
  - **U.MNO-OTA and U.MNO-SD, with security attribute D.MNO_KEYS**
- o **information: transmission of commands**.

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

- o **The TOE shall permit communication between U.MNO-OTA and U.MNO-SD in a SCP80 or SCP81 secure channel**.

FDP_IFF.1.3/SCP The TSF shall enforce the **None**.

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

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

- o **The TOE shall reject communication between U.SM-DPplus and S.ISD-R if it is not performed in a SCP-SGP22 secure channel**.

Application Note:

More details on the secure channels can be found in [24]

- For SM-DP+: §5.5
- For MNO-SD: §5.4

FTP_ITC.1/SCP Inter-TSF trusted channel

FTP_ITC.1.1/SCP 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/SCP The TSF shall permit **another trusted IT product** to initiate communication via the trusted channel.

FTP_ITC.1.3/SCP The TSF shall initiate communication via the trusted channel for **functions listed above with the secure channels SCP80, SCP81, SCP02, SCP03 et SCP03t.**

The TSF permits remote actors to initiate communication via a trusted channel in the following cases:

- o **The TSF shall permit the SM-DP+ to open a SCP-SGP22 (SCP02, SCP03 et SCP03t) secure channel to transmit the following operations:**
  - **ES8+.InitialiseSecureChannel**
  - **ES8+.ConfigureISDP**

- **ES8+.StoreMetadata**
- **ES8+.ReplaceSessionKeys**
- **ES8+.LoadProfileElements.**

  - **SCP80 is provided to build secure channels to MNO OTA Platform (chapter 5.4 of [24]). The TSF also permit to use a SCP81 secure channel to perform the same functions than the SCP80 secure channel. The TSF permits the remote OTA Platform to open a SCP80 or SCP81 secure channel to transmit the following operation:**
    - **ES6.UpdateMetadata.**
  - **The TSF shall permit the LPAd to transmit the following operations:**
    - **ES10a.GetEuiccConfiguredAddresses**
    - **ES10a.SetDefaultDpAddress**
    - **ES10b.PrepareDownload**
    - **ES10b.LoadBoundProfilePackage**
    - **ES10b.GetEUICCChallenge**
    - **ES10b.GetEUICCInfo**
    - **ES10b.ListNotification**
    - **ES10b.RetrieveNotificationsList**
    - **ES10b.RemoveNotificationFromList**.

Application Note:
The ST includes the requirements stated by [24]:

- The secure channels to SM-DP+ is SCP-SGP22 secure channels. Identification of endpoints is addressed by the use of AES according to [11] using the parameters defined in [24], chapters 2.6 and 5.5.
- SCP80 is provided to build secure channels to MNO OTA Platform (chapter 5.4 of [24]). The TSF also permit to use a SCP81 secure channel to perform the same functions than the SCP80 secure channel.

Related keys are:

- either generated on-card (D.SECRETS); see FCS_CKM.1/SCP-SM for further details,
- or distributed along with the profile (D.MNO_KEYS); see FCS_CKM.2/SCP-SM-MNO for further details. In terms of commands, the TSF shall permit remote actors to initiate communication via a trusted channel in the following cases:
- The TSF shall permit the SM-DP+ to open a SCP-SGP22 secure channel to transmit the following operations:
  - ES8+.InitialiseSecureChannel
  - ES8+.ConfigureISDP
  - ES8+.StoreMetadata
  - ES8+.ReplaceSessionKeys
  - ES8+.LoadProfileElements.
- The TSF shall permit the remote OTA Platform to open a SCP80 or SCP81 secure channel to transmit the following operation:
  - ES6.UpdateMetadata.

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

FDP_ITC.2.1/SCP The TSF shall enforce the **Secure Channel Protocol information flow control SFP** when importing user data, controlled under the SFP, from outside of the TOE.

FDP_ITC.2.2/SCP The TSF shall use the security attributes associated with the imported user data.

FDP_ITC.2.3/SCP 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/SCP 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/SCP The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: **None**.

FPT_TDC.1/SCP Inter-TSF basic TSF data consistency

FPT_TDC.1.1/SCP The TSF shall provide the capability to consistently interpret

- **Commands from U.SM-DPplus and U.MNO-OTA**
- **Downloaded objects from U.SM-DPplus and U.MNO-OTA**

when shared between the TSF and another trusted IT product.

FPT_TDC.1.2/SCP The TSF shall use **the rules defined in the GP [39] section 11.8** when interpreting the TSF data from another trusted IT product.

Application Note:
The commands related to the SFRs FPT_TDC.1/SCP, FDP_IFC.1/SCP, FDP_IFF.1/SCP and the Downloaded objects related to this SFR FPT_TDC.1/SCP are listed below:

- SM-DP+ commands
  - ES8+.InitialiseSecureChannel
  - ES8+.ConfigureISDP
  - ES8+.StoreMetadata
  - ES8+.ReplaceSessionKeys
  - ES8+.LoadProfileElements
- Downloaded objects from SM-DP+
  - Session keys
  - Profile Metadata (including PPR data)
- MNO commands
  - ES6.UpdateMetadata
- Downloaded objects from MNO OTA Platform
  - Profile Metadata (including PPR data).

FDP_UCT.1/SCP Basic data exchange confidentiality

FDP_UCT.1.1/SCP The TSF shall enforce the **Secure Channel Protocol information flow control SFP** to **receive** user data in a manner protected from unauthorised disclosure.

Application Note:
This SFR is related to the protection of:

- Profiles downloaded from SM-DP+.

Related keys are:

- either generated on-card (D.SECRETS): see FCS_CKM.1/SCP-SM for further details;
- or distributed along with the Profile (D.MNO_KEYS); see FCS_CKM.2/SCP-MNO for further details.

FDP_UIT.1/SCP Data exchange integrity

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

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

Application Note:
This SFR is related to the protection of:

- Profiles downloaded from SM-DP+;
- Commands received from SM-DP+ and MNO OTA Platform;
- PPR received from the MNO OTA Platform. Related keys are:
    - either generated on-card (D.SECRETS): see FCS_CKM.1/SCP-SM for further details;
    - or distributed along with the Profile (D.MNO_KEYS); see FCS_CKM.2/SCP-MNO for further details.

FCS_CKM.1/SCP-SM Cryptographic key generation

FCS_CKM.1.1/SCP-SM The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **ElGamal elliptic curves key agreement (ECKA)** and specified cryptographic key sizes **256** that meet the following: **ECKA-EG using one of the following standards:**

- **NIST P-256 (FIPS PUB 186-3 Digital Signature Standard),**
- **brainpoolP256r1 (BSI TR-03111, Version 1.11, RFC 5639)**.

Application Note:
This key generation mechanism is used to generate

- D.SECRETS keyset via the ES8+.InitialiseSecureChannel command, using the U.SM-DPplus public key otPK.DP.ECKA.

FCS_CKM.2/SCP-MNO Cryptographic key distribution

FCS_CKM.2.1/SCP-MNO The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method **Key distribution and key derivation scheme for 5G for network nodes** that meets the following: **As defined in [47]**.

Application Note:
This SFR is related to the distribution of

- D.MNO_KEYS during profile download. Note: this SFR does not apply to the private keys loaded pre-issuance of the TOE (D.SK.EUICC.ECDSA).

FCS_CKM.4/SCP-SM Cryptographic key destruction

FCS_CKM.4.1/SCP-SM The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method **the keys are reset with the method clearKey()** that meets the following: **"Java Card API" specification [32]. The methods 'reset' and 'setKeyFormat' call the method key.clearKey() for clearing the value of each key**.

Application Note:
This SFR is related to the destruction of the following keys:

- D.SECRETS
- CERT.DPauth.ECDSA
- CERT.DPpb.ECDSA
- CERT.DP.TLS
- D.CERT.EUICC.ECDSA
- D.SK.EUICC.ECDSA
- D.PK.CI.ECDSA.

FCS_CKM.4/SCP-MNO Cryptographic key destruction

FCS_CKM.4.1/SCP-MNO The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method **the keys are reset with the method clearKey()** that meets the following: **"Java Card API" specification [32]. The methods 'reset' and 'setKeyFormat' call the method key.clearKey() for clearing the value of each key**.

Application Note:
This SFR is related to the destruction of the following keys:

- D.MNO_KEYS.

8.1.2.3 Security Domains

This package describes the specific requirements applicable to the Security Domains belonging to the TOE. In particular it defines:

- The rules under which the S.ISD-R can perform its functions (ISD-R access control SFP in FDP_ACC.1/ISDR and FDP_ACF.1/ISDR),
- The rules under which the S.ISD-R can perform ECASD functions and obtain output data from these functions (ECASD access control SFP in FDP_ACC.1/ECASD and FDP_ACF.1/ECASD).

FDP_ACC.1/ISDR Subset access control

FDP_ACC.1.1/ISDR The TSF shall enforce the **ISD-R access control SFP** on

- **subjects: S.ISD-R**
- **objects: S.ISD-P**
- **operations:**
  - **Create and configure profile**
  - **Store profile metadata**
  - **Enable profile**
  - **Disable profile**
  - **Delete profile**
  - **Perform a Memory reset**.

Application Note:
This policy describes the rules to be applied to access Platform Management operations. It covers the access to operations by ISD-R required by sections 5.x of [24].

FDP_ACF.1/ISDR Security attribute based access control

FDP_ACF.1.1/ISDR The TSF shall enforce the **ISD-R access control SFP** to objects based on the following:

- **subjects: S.ISD-R**
- **objects:**
  - **S.ISD-P with security attributes "state" and "PPR"**
- **operations:**
  - **Create and configure profile**
  - **Store profile metadata**
  - **Enable profile**
  - **Disable profile**
  - **Delete profile**
  - **Perform a Memory reset**.

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

- **Enabling a S.ISD-P is authorized only if**
  - **the corresponding S.ISD-P is in the state "DISABLED" and**
  - **the currently enabled S.ISD-P's PPR data allows its disabling.**
- **Disabling a S.ISD-P is authorized only if**
  - **the corresponding S.ISD-P is in the state "ENABLED" and**
  - **the corresponding S.ISD-P's PPR data allows its disabling.**
- **Deleting a S.ISD-P is authorized only if**
  - **the corresponding S.ISD-P is not in the state "ENABLED" and**

  - **the corresponding S.ISD-P's PPR data allows its deletion.**
- **Performing a S.ISD-P Memory reset is authorized regardless of the involved S.ISD-P's state or PPR attribute**.

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

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

Application Note:
This policy describes the rules to be applied to access Platform Management or eUICC Management operations. It covers the access to the following operations by ISD-R required by sections 5.x of [24]:

- ES8+.ConfigureISDP (Create and configure profile)
- ES8+.StoreMetadata (Store profile metadata)
- ES10c.EnableProfile (Enable profile)
- ES10c.DisableProfile (Disable profile)
- ES10c.DeleteProfile (Delete profile)
- ES10c.eUICCMemoryReset (Perform a Memory reset).

FDP_ACC.1/ECASD Subset access control

FDP_ACC.1.1/ECASD The TSF shall enforce the **ECASD access control SFP** on **subjects: S.ISD-R, objects: S.ECASD, operations:**

- **execution of a ECASD function**
- **access to output data of these functions,**

No additional subjects, objects, or operations.

FDP_ACF.1/ECASD Security attribute based access control

FDP_ACF.1.1/ECASD The TSF shall enforce the **ECASD access control SFP** to objects based on the following:

- **subjects: S.ISD-R, with security attribute i§AIDi¨ objects: S.ECASD operations:**
  - **execution of a ECASD function**
    - **Verification of the off-card entities Certificates (SM-DP+, SM-DS), provided by an ISD-R, with the CI public key (PK.CI.ECDSA)**
    - **Creation of an eUICC signature on material provided by an ISD-R**
  - **access to output data of these functions.**
- **None**.

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

- **Authorized users: only S.ISD-R, identified by its AID, shall be authorized to execute the following S.ECASD functions:**

- **Verification of a certificate CERT.DPauth.ECDSA, CERT.DPpb.ECDSA, CERT.DP.TLS, CERT.DSauth.ECDSA, or CERT.DS.TLS, provided by an ISD-R, with the CI public key (PK.CI.ECDSA)**
- **Creation of an eUICC signature, using D.SK.EUICC.ECDSA, on material provided by an ISD-R.**

o **None**.

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

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

Application Note:
The APDU for certificate verification is locked in use phase.

8.1.2.4 Platform Services

This package describes the specific requirements applicable to the Profile Policy Enabler, Profile Package Interpreter and the Telecom Framework. In particular it defines:

- FDP_IFC.1/Platform_services and FDP_IFF.1/Platform_services: the measures taken to control the flow of information between the Security Domains and PPE, PPI or Telecom Framework;
- FPT_FLS.1/Platform_services: the measures to enforce a secure state in case of failures of PPE, PPI or Telecom Framework.

FDP_IFC.1/Platform_services Subset information flow control

FDP_IFC.1.1/Platform_services The TSF shall enforce the **Platform services information flow control SFP** on **users/subjects:**

- **S.ISD-R, S.ISD-P, U.MNO-SD**
- **Platform code (S.PPE, S.PPI, S.TELECOM) information:**
- **D.PROFILE_NAA_PARAMS**
- **D.PROFILE_POLICY_RULES**
- **D.PLATFORM_RAT operations:**
- **installation of a profile**
- **PPR and RAT enforcement**
- **network authentication**.

FDP_IFF.1/Platform_services Simple security attributes

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

- **S.ISD-R, S.ISD-P, U.MNO-SD, with security attribute "application identifier (AID)" information:**
- **D.PROFILE_NAA_PARAMS**

- o **D.PROFILE_POLICY_RULES**
- o **D.PLATFORM_RAT operations:**
- o **installation of a profile**
- o **PPR and RAT enforcement**
- o **network authentication**.

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

- o **D.PROFILE_NAA_PARAMS shall be transmitted only:**
  - **by U.MNO-SD to S.TELECOM in order to execute the network authentication function**
  - **by S.ISD-R to S.PPI using the profile installation function**
- o **D.PROFILE_POLICY_RULES shall be transmitted only**
  - **by S.ISD-R to S.PPE in order to execute the PPR enforcement function**
- o **D.PLATFORM_RAT shall be transmitted only**
  - **by S.ISD-R to S.PPE in order to execute the RAT enforcement function**.

FDP_IFF.1.3/Platform_services The TSF shall enforce the **none.**

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

FDP_IFF.1.5/Platform_services The TSF shall explicitly deny an information flow based on the following rules: **the rules describing the communication protocol used by the MNO-SD and the card for transmitting a new profile. This dispatcher is only accessible through ISD-R or ISD-P and associated communication.**

FPT_FLS.1/Platform_services Failure with preservation of secure state

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

- o **failure that lead to a potential security violation during the processing of a S.PPE, S.PPI or S.TELECOM API specific functions:**
  - **Installation of a profile**
  - **PPR and RAT enforcement**
  - **Network authentication**
- o **No other type of failure**.

8.1.2.5 Security management

This package includes several supporting security functions:

- Random number generation (FCS_RNG.1)
- User data and TSF self-protection measures:
  - o TOE emanation (FPT_EMS.1)
  - o protection from integrity errors (FDP_SDI.1)

- residual data protection (FDP_RIP.1)
- preservation of a secure state (FPT_FLS.1/VM)

- Security management measures:
  - Management of security attributes such as Platform data (FMT_MSA.1/PLATFORM_DATA), PPR (FMT_MSA.1/PPR), (FMT_MSA.1/RAT) and keys (FMT_MSA.1/CERT_KEYS) with restrictive default values (FMT_MSA.3);
  - Management of roles and security functions (FMT_SMR.1/EUICC and FMT_SMF.1/EUICC).

FPT_EMS.1 TOE Emanation

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

- **D.SECRETS;**
- **D.SK.EUICC.ECDSA**

and **the secret keys which are part of the following keysets:**

- **D.MNO_KEYS,**
- **D.PROFILE_NAA_PARAMS**

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

- **D.SECRETS;**
- **D.SK.EUICC.ECDSA**

and **the secret keys which are part of the following keysets:**

- **D.MNO_KEYS,**
- **D.PROFILE_NAA_PARAMS**

Application Note:
The TOE shall prevent attacks against the secret data of the TOE where the attack is based on external observable physical phenomena of the TOE. Such attacks may be observable at the interfaces of the TOE or may originate from internal operation of the TOE or may originate from an attacker that varies the physical environment under which the TOE operates. The set of measurable physical phenomena is influenced by the technology employed to implement the TOE. Examples of measurable phenomena are variations in the power consumption, the timing of transitions of internal states, electromagnetic radiation due to internal operation, radio emission. Due to the heterogeneous nature of the technologies that may cause such emanations, evaluation against state-of-the-art attacks applicable to the technologies employed by the TOE is assumed. Examples of such attacks are, but are not limited to, evaluation of TOE's electromagnetic radiation, simple power analysis (SPA), differential power analysis (DPA), timing attacks, and so on.

FDP_SDI.1 Stored data integrity monitoring

FDP_SDI.1.1 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**.

Application Note:
Refinement: The notion of integrity-sensitive data covers the assets of the Security Target TOE that

require to be protected against unauthorized modification, including but not limited to the assets of this ST that require to be protected against unauthorized modification:

- D.MNO_KEYS
- Profile data
  - D.PROFILE_NAA_PARAMS
  - D.PROFILE_IDENTITY
  - D.PROFILE_POLICY_RULES
  - D.PROFILE_USER_CODES
- Management data
  - D.PLATFORM_DATA
  - D.DEVICE_INFO
  - D.PLATFORM_RAT
- Identity management data
  - D.SK.EUICC.ECDSA
  - D.CERT.EUICC.ECDSA
  - D.PK.CI.ECDSA
  - D.EID
  - D.SECRETS
  - D.CERT.EUM.ECDSA
  - D.CRLs if existing

FDP_RIP.1/EUICC Subset residual information protection

FDP_RIP.1.1/EUICC The TSF shall ensure that any previous information content of a resource is made unavailable upon the **allocation of the resource to and deallocation of the resource from** the following objects:

- **D.SECRETS;**
- **D.SK.EUICC.ECDSA;**
- **The secret keys which are part of the following keysets:**
  - **D.MNO_KEYS,**
  - **D.PROFILE_NAA_PARAMS**.

FPT_FLS.1/EUICC Failure with preservation of secure state

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

- **failure of creation of a new ISD-P by ISD-R**
- **failure of installation of a profile by ISD-R**.

FMT_MSA.1/PLATFORM_DATA Management of security attributes

FMT_MSA.1.1/PLATFORM_DATA The TSF shall enforce the **ISD-R access control policy** to restrict the ability to **modify** the security attributes **following parts of D.PLATFORM_DATA:**

- **ISD-P state**

to

- **S.ISD-R to modify ISD-P state**
  - **from "INSTALLED" to "SELECTABLE" (during ISD-P creation)**
  - **from "ENABLED" to "DISABLED" (during profile disabling)**
- **S.ISD-R to modify ISD-P state**
  - **from "DISABLED" to "ENABLED" (during profile enabling)**.

FMT_MSA.1/PPR Management of security attributes

FMT_MSA.1.1/PPR The TSF shall enforce the **Security Channel protocol information flow SFP, ISD-P content access control SFP and ISD-R access control SFP** to restrict the ability to **change_default, query, modify and delete** the security attributes

- **D.PROFILE_POLICY_RULES**

to

- **S.ISD-R to change_default, via function "ES8.ConfigureISDP"**
- **S.ISD-R to query**
- **S.ISD-P to modify, via function "ES6.UpdateMetadata"**
- **S.ISD-R to delete, via function "ES10c.DeleteProfile"**.

FMT_MSA.1/CERT_KEYS Management of security attributes

FMT_MSA.1.1/CERT_KEYS The TSF shall enforce the **Security Channel protocol information flow SFP, ISD-R access control SFP and ECASD access control SFP** to restrict the ability to **query and delete** the security attributes

- **D.CERT.EUICC.ECDSA**
- **D.PK.CI.ECDSA**
- **D.CERT.EUM.ECDSA**
- **D.MNO_KEYS**

to

- **S.ISD-R for:**
  - **query D.PK.CI.ECDSA**
  - **delete D.MNO_KEYS, via function "ES10c.DeleteProfile"**
- **no actor for other operations**.

Application Note:
The modification of D.MNO_KEYS keysets is forbidden. To modify the keysets, one must delete the profile and load another profile.

FMT_SMF.1/EUICC Specification of Management Functions

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

- **ISD-R access control**
- **Security Channel protocol information flow control (for roles: S.ISD_R and S.ISD_P)**
- **ISD-P content access control,**
- **ECASD access control (for role S.ECASD and S.ISD-R),**
- **Platform services information flow control (for roles S.ISD_R and S.PPE)**.

FMT_SMR.1/EUICC Security roles

FMT_SMR.1.1/EUICC The TSF shall maintain the roles

- **External users:**
  - **U.SM-DPplus**
  - **U.MNO-SD**
  - **U.MNO-OTA**
- **Subjects:**
  - **S.ISD-R**
  - **S.ISD-P**
  - **S.ECASD**
  - **S.PPI**
  - **S.PPE**
  - **S.TELECOM.**.

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

Application Note:
The roles defined here correspond to the users and subjects defined in this ST.

FMT_MSA.1/RAT Management of security attributes

FMT_MSA.1.1/RAT The TSF shall enforce the **Platform services information flow SFP and ISD-R access control SFP** to restrict the ability to **query** the security attributes

- **D.PLATFORM_RAT**

to

- **S.ISD-R to query**
- **S.PPE to query**.

FMT_MSA.3 Static attribute initialisation

FMT_MSA.3.1 The TSF shall enforce the **Security Channel Protocol information flow control SFP, ISD-P content access control SFP, ISD-R access control SFP and ECASD access control SFP** to provide **restrictive** default values for security attributes that are used to enforce the SFP.

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

8.1.2.6 Mobile Network authentication

This package defines the requirements related to the authentication of the eUICC on MNO networks. The TSF must implement cryptographic mechanisms for the authentication on the MNO network (FCS_COP.1/Mobile_network) and manage the keys securely (FCS_CKM.2/Mobile_network and FCS_CKM.4/Mobile_network).

FCS_COP.1/Mobile_network Cryptographic operation

FCS_COP.1.1/Mobile_network The TSF shall perform **Network authentication** in accordance with a specified cryptographic algorithm **MILENAGE, Tuak no other algorithm** and cryptographic key sizes **according to the corresponding standard** that meet the following:

- **MILENAGE according to standard [20] with the following restrictions:**
  - **Only use 128-bit AES as the kernel function- do not support other choices**
  - **Allow any value for the constant OP**
  - **Allow any value for the constants C1-C5 and R1-R5, subject to the rules and recommendations in section 5.3 of the standard [20]**
- **Tuak according to [21] with the following restrictions:**
  - **Allow any value of TOP**
  - **Allow multiple iterations of Keccak**
  - **Support 256-bit K as well as 128-bit**
  - **To restrict supported sizes for RES, MAC, CK and IK to those currently supported in 3GPP standards**.

FCS_CKM.2/Mobile_network Cryptographic key distribution

FCS_CKM.2.1/Mobile_network The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method **Key distribution and key derivation scheme for 2G/3G/4G/5G network nodes** that meets the following: **As defined in [20] [47]**.

Application Note:
The keys in this SFR are the Mobile Network authentication keys included in the asset D.PROFILE_NAA_PARAMS. These keys are distributed as a part of the MNO profile during profile download.

FCS_CKM.4/Mobile_network Cryptographic key destruction

FCS_CKM.4.1/Mobile_network The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method **the keys are reset with the method clearKey()** that meets the following: **"Java Card API" specification [32]. The methods 'reset' and 'setKeyFormat' call the method key.clearKey() for clearing the value of each key**.

8.1.2.7 SUCI

FCS_CKM.1/SUCI Cryptographic key generation

FCS_CKM.1.1/SUCI The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **ECC ephemeral key** and specified cryptographic key sizes **256 bits** that meet the following:

- **X25519 EC Diffie-Hellman primitive IEEE Std 1363a-2004 [IEEE],**
- **Elliptic Curve Cofactor Diffie-Hellman Primitive [45]**.

Application Note:
The processing on UE side shall be done according to the encryption operation defined in [44]. This SFR implicitly contains the requirements for the hashing functions used for key derivation by demanding compliance to ANSI-X9.63-KDF.

FCS_CKM.4/SUCI Cryptographic key destruction

FCS_CKM.4.1/SUCI The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method **the keys are reset with the method clearKey()** that meets the following: **"Java Card API" specification [32]. The methods 'reset' and 'setKeyFormat' call the method key.clearKey() for clearing the value of each key**.

Application Note:
The keys are reset as specified in [32] Key class, with the method clearKey(). Any access to a cleared key for ciphering or signing shall throw an exception. This SFR is related to the key destruction of K ephemeral key.

FCS_COP.1/SUCI Cryptographic operation

FCS_COP.1.1/SUCI The TSF shall perform **see table below** in accordance with a specified cryptographic algorithm **see table below** and cryptographic key sizes **see table below** that meet the following: **see table below for encryption with signature**

Cryptographic operation	**Algorithm**	**Key length**	**Standard**
Encrytpion Ephemeral derived key (K)	**AES with CTR Mode**	**128**	**FIPS 197, NIST SP800 38A (CTR mode)**

Key agreement for profile A protection scheme	**ECDH Curve25519**	**256 bits**	**IEEE P1363**
Key agreement for profile B protection scheme	**ECDH Secp256r1**	**256 bits**	**IEEE P1363**
Data integrity mechanism	**HMAC**	**SHA-256**	**FIPS 198-1 (HMAC)**

Application Note:
Current Implementation supports Profile A and B defined in 3GPP TS 33.501 [44]. The final output shall be the concatenation of the ECC ephemeral public key, the ciphertext value, the MAC tag value, and any other parameters, if applicable; Profile A (Curve25519) and Profile B (SECP256r1).

8.2 Security Assurance Requirements

The Evaluation Assurance Level is EAL4 augmented with AVA_VAN.5, ALC_DVS.2 and ALC_FLR.1.

8.3 Security Requirements Rationale

8.3.1 Objectives

8.3.1.1 Security Objectives for the TOE

Java Card

IDENTIFICATION

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/ADEL, FMT_SMF.1/CM, 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).

EXECUTION

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), 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/Firewall, FMT_SMF.1/Firewall, 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, S, FMT_MSA.1/JCRE, FMT_MSA.1/JCVM,) also indirectly contribute to meet this objective.

This objective is also covered by the following additional SFRs:

- o Stack control (*/RV_Stack): FDP_ACC.2/RV_Stack, FDP_ACF.1/RV_Stack, FMT_MSA.1/RV_Stack, FMT_MSA.2/RV_Stack, FMT_MSA.3/RV_Stack, FMT_SMF.1/RV_Stack

- Heap control (*/RV_Heap): FDP_ACC.2/RV_Heap, FDP_ACF.1/RV_Heap, FMT_MSA.1/RV_Heap, FMT_MSA.2/RV_Heap, FMT_MSA.3/RV_Heap, FMT_SMF.1/RV_Heap
- Transient control (*/RV_Transient): FDP_ACC.2/RV_Transient, FDP_ACF.1/RV_Transient, FMT_MSA.1/RV_Transient, FMT_MSA.2/RV_Transient, FMT_MSA.3/RV_Transient, FMT_SMF.1/RV_Transient

For each of those control, the SFR define the access control (FDP_ACC and FDP_ACF), the operation (FMT_MSA) and the role (FMT_SMF).

The Stack control enforces O.FIREWALL by defining additional rules, such as the control of the stack is more precise. Information is provided in the application note.

The Heap control enforces O.FIREWALL by defining additional rules, such as the heap usage is improved. Information is provided in the application note.

The Transient enforces O.FIREWALL by defining additional rules, such as the heap usage is improved. Information is provided in the application note.

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. If the TOE provides JCRMI functionality, protection of the array parameters of remotely invoked methods, which are global as well, is covered by the general initialization of method parameters (FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL and FDP_RIP.1/TRANSIENT).

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 or to the global byte array of the applet's install method. Such a pointer could be used to access and modify it when the buffer is being used by another application.

O.NATIVE This security objective is covered by FDP_ACF.1/FIREWALL: the only means to execute native code is the invocation of a Java Card API method. This objective mainly relies on the environmental objective OE.CAP_FILE, which uphold the assumption A.CAP_FILE.

O.REALLOCATION This security objective is satisfied by the following SFRs: FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray, FDP_RIP.1/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/TRANSIENT, FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/ADEL, which imposes that the contents of the re-allocated block shall always be cleared before delivering the block.

O.RESOURCES The TSFs detects stack/memory overflows during execution of applications (FAU_ARP.1, FPT_FLS.1/ADEL, FPT_FLS.1/VM, FPT_FLS.1/ODEL, FPT_FLS.1/Installer). Failed installations are not to create memory leaks (FDP_ROL.1/FIREWALL, FPT_RCV.3/Installer) as well. Memory management is controlled by the TSF (FMT_MTD.1/JCRE, FMT_MTD.3/JCRE, FMT_SMR.1/Installer, FMT_SMR.1/Firewall, FMT_SMF.1/Firewall, FMT_SMR.1/ADEL, FMT_SMF.1/ADEL, FMT_SMF.1/CM and FMT_SMR.1/CM).

O.ARRAY_VIEWS_CONFID Array views have security attributes of temporary objects where the JCVM information flow control policy (FDP_IFF.1/JCVM, FDP_IFC.1/JCVM) prevents an application from storing a reference to the array view. Furthermore, array views may not have

ATTR_READABLE_VIEW security attribute which ensures that no application can read the contents of the array view.

O.ARRAY_VIEWS_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.

SERVICES

O.ALARM This security objective is met by FPT_FLS.1/Installer, FPT_FLS.1/VM, 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/CM-SCP, FCS_CKM.4/CM-SCP, FCS_CKM.5/KDF, FCS_COP.1/Disp and FCS_COP.1/Patch. FPR_UNO.1 and FCS_CKM.4/SUCI, FCS_COP.1/SUCI 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/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 and FDP_SDI.2/DATA as well. Precisely it is met by the following components: FCS_CKM.1/CM-SCP, FCS_CKM.4/CM-SCP, FCS_CKM.5/KDF, FCS_CKM.4/SUCI, FCS_COP.1/Disp, FCS_COP.1/Patch, FPR_UNO.1, FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray FDP_RIP.1/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL and FDP_RIP.1/TRANSIENT.

O.PIN-MNGT This security objective is ensured by FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray, FDP_RIP.1/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL, FDP_RIP.1/TRANSIENT, FPR_UNO.1, FDP_ROL.1/FIREWALL and FDP_SDI.2/DATA security functional requirements. The TSFs behind these are implemented by API classes. The firewall security functions FDP_ACC.2/FIREWALL and FDP_ACF.1/FIREWALL shall protect the access to private and internal data of the objects.

O.TRANSACTION Directly met by FDP_ROL.1/FIREWALL, FDP_RIP.1/ABORT, FDP_RIP.1/ODEL, FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray, FDP_RIP.1/bArray, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL, FDP_RIP.1/TRANSIENT and FDP_RIP.1/OBJECTS (more precisely, by the element FDP_RIP.1/ABORT).

O.RNG This security objective is directly covered by FCS_RNG.1 which ensures the cryptographic quality of random number generation.

OBJECT DELETION

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.

APPLET MANAGEMENT

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, 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).

Additional security objectives for the TOE

O.SCP.SUPPORT The components FPT_RCV.4/SCP (SCP stands for smart card platform) are used to support the objective O.SCP.SUPPORT to assist the TOE to recover in the event of a power failure. If the power fails or the card is withdrawn prematurely from the CAD the operation of the TOE may be interrupted leaving the TOE in an inconsistent state.

All the Crypto SFRS support this objective as they provide secure low-level cryptographic processing to the Java Card System and Global Platform:

- FCS_CKM.1/CM-SCP, FCS_CKM.4/CM-SCP, FCS_COP.1/Disp and FCS_COP.1/Patch,
- FCS_COP.1/CM-SCP,

All the FSRs related to the Firewall contribute to the realization of the objective.

The FDP_ROL.1 Firewall ensures the rollback of some operations within the specified scope as defined in the ROL.1.2/Firewall.

Application Note: all SFRs related to O.OPERATE and O.ALARM support the O.SCP.SUPPORT

O.SCP.IC This objective is met by the component FPR_UNO.1 and FPT_EMS.1 for the IC resistance and FCS_RNG.1 for RNG quality

O.SCP.RECOVERY The component FPT_RCV.4/SCP is used to support the objective O.SCP.RECOVERY to assist the TOE to recover in the event of a power failure. If the power fails or the card is withdrawn prematurely from the CAD the operation of the TOE may be interrupted leaving the TOE in an inconsistent state. This objective is met by the components FPT_FLS.1/VM and FAU_ARP.1.

O.CARD_MANAGEMENT This objective is fulfilled by the following set of SFR:

The FDP_ACC.2/ADEL and FDP_ACF.1/ADEL contribute to meet the ADEL access control policy that ensures the non-introduction of security holes.

The FDP_RIP.1/ADEL ensure that the deleted information is not accessible.

The FMT_MSA.1/ADEL ensures restrict the ability to modify the secure attributes the FMT_MSA.3/ADEL ensures the assignment of restrictive values.

The FMT_SMR.1/ADEL maintains the role of the applet deletion manager.

The FPT_FLS.1/ADEL contributes to the objective by protecting the TSFs against possible failures of the deletion procedure.

The 2 SFRs FPT_RCV.3/Installer and FPT_FLS.1/Installer contributes to meet the objective by protecting the TSFs from failures of the deletion procedure.

The SFR FDP_UIT.1/CM contributes by enforcing the Secure Channel Protocol Information flow control policy and the Security Domain access control policy which control the integrity of the corresponding data.

The SFR FIA_UID.1/CM testes if the Secure Channel is open to allow card management operations.

The SFR FDP_IFF.1/CM ensures the access control policy for the loaded data (as packages).

The FCO_NRO.2/CM_DAP this SFR ensures the origin of the load file. It verifies the identity of the origin of the load file before start the loading.

FCO_NRO.2/CM_DAP this SFR generates an evidence of the origin of the transmitted load file during CAP File loading.

The FDP_IFC.2/CM, this SFR ensures that loading commands are issued in the Secure Channel session.

The SFR FDP_ROL.1/FIREWALLensures that the card management operations are cleaned aborted.

The SFR FDP_ITC.2/Installer enforces the Firewall access control policy and flow control policy when importing card management data.

The SFR FPT_FLS.1/ODEL ensures the preservation of secure state when failures occur.

The SFR FMT_MSA.1/CM ensures the management of the security attributes to the card manager, for the modification of the life cycle of the card, the keyset version and value,…

The SFR FMT_MSA.3/CM, this SFR ensures that the security attributes can only be changed by the card manager.

The SFR FMT_SMF.1/CM Only the card manger is able to modify the security attributes of the management functions. The security role is specified in the FMT_SMR.1/CM.

The SFR FPT_TDC.1/CM ensure that key sets and packages loaded are well under key management.

The SFR FTP_ITC.1/CM ensures the trusted Channel Communications.

FIA_UAU.1/CM, FIA_UAU.4/CardIssuer ensure the authentication of the card issuer before gaining access to management operations.

The FPR_UNO.1 ensures the un-observability of the CM key when imported..

The FPT_TST.1 This TSF contributes to ensure the correct operation of the card management functions as it tests the integrity of the TSF functions during initial start-up.

The SFR FPT_TDC.1/CM ensures that key sets and packages loaded are well under key management.

O.PATCH_LOADING Authentication of the entity loading the patch by the TOE

FDP_ACC.2/Patch, FDP_ACF.1/Patch provide access control for patch loading. The subject entitled to load the patch is authenticated by the TOE thanks to FCS_COP.1/Patch.

Authentication of the TOE

To avoid impersonation of the TOE by a fake chip, the TOE authenticates itself; from phase 6 (after patch loading) with FTP_ITC.1/CM and FCS_COP.1/CM-SCP thanks to the TOE authentication key (ISK/KMC). From phase 6, the TOE authentication is required prior to any trusted channel establishment with FTP_ITC.1/CM (data sent by the TOE must be decrypted to carry on the authentication).

The patch is also included in TOE identification, each patch identified is supported by SFR linked to O.SID.

Integrity, confidentiality and authenticity of the patch during loading

Patch loading is performed in a confidential manner with FDP_UCT.1/Patch and protected in integrity and confidentiality with FDP_UIT.1/Patch. Confidentiality, integrity and authenticity of the patch loading is supported by cryptographic mechanisms supported by FCS_COP.1/Patch.

Patch data to be written in the TOE have been prior encrypted by the TOE developer using JSK key. Once these data loaded, the integrity (SHA256) of the modified code is update and compared to the provided one in the patch package. finally the access control for the importation of patch is ensured by FDP_ITC.1/Patch

Irreversible locking of the patch loading features

The patch can be loaded during the TOE's life cycle.

Identification of the patch after loading

Once loaded and during the rest of the TOE life cycle, the identification and authentication (unique identifier of the patch) of the patch, being a part of the TOE is provided by FAU_STG.2/Patch. When requested, the identification and authentication data (of entire code, including patch) are dynamically retrieved from the patch code stored in the non-volatile memory of the TOE.

Integrity check before usage of the patch

At start up, the integrity of the entire code, patch included, is checked by the TOE through self-tests provided by FPT_TST.1. In case the computed signature differs from the one stored in NVM, an integrity error is detected and a killcard is raised.

Additional objective for Sensitive Array package

O.SENSITIVE_ARRAYS_INTEG This objective is fulfilled by FDP_SDI.2/ARRAY. It ensures ensures that integrity errors related to the user data stored in sensitive arrays are detected by the TOE..

Consumer Device

eUICC proof of identity

O.PROOF_OF_IDENTITY This objective is covered by the extended requirement FIA_API.1/Authentication Proof of Identity and FCS_CKM.1/SUCI for key generation.

Platform services

O.OPERATE The TOE is protected in various ways against applets' actions (FPT_TDC.1/VM), FPT_FLS.1/EUICC requires that failures do not impact on the security of the TOE.

O.API FDP_IFC.1/Platform_services Subset information flow control, FDP_IFF.1/Platform_services Simple security attributes, FMT_MSA.3/Static attribute initialisation, FMT_SMF.1/EUICC and FMT_SMR.1/EUICC state the policy for controlling the access to TOE services and resources by the Application Layer. Atomicity is provided by the FPT_FLS.1/Platform_services Failure with preservation of secure state.

Data protection

O.DATA-CONFIDENTIALITY FDP_UCT.1/SCP Basic data exchange confidentiality addresses the reception of data from off-card actors, while the access control SFRs (FDP_ACC.1/ISDR Subset access control, FDP_ACC.1/ECASD Subset access control) address the isolation between Security Domains. FPT_EMS.1/TOE Emanation ensures that secret data stored or transmitted within the TOE shall not be disclosed in cases of side channel attacks. FDP_RIP.1/EUICC /Subset residual information protection ensures that no residual confidential data is available. FCS_COP.1/Mobile_network Cryptographic operation, FCS_CKM.2/Mobile_network Cryptographic key distribution, and FCS_CKM.4/Mobile_network Cryptographic key destruction address the cryptographic algorithms present in the Telecom Framework, the distribution and the destruction of associated keys.

O.DATA-INTEGRITY FDP_UIT.1/SCP Data exchange integrity addresses the reception of data from off-card actors, while the access control SFRs (FDP_ACC.1/ISDR Subset access control, FDP_ACC.1/ECASD Subset access control) address the isolation between Security Domains. FDP_SDI.1/Stored data integrity monitoring specifies the Profile data that is monitored in case of an integrity breach (for example modification of the received profile during the installation operation). FPT_TST.1 would contribute to the protection of integrity.

Connectivity

O.ALGORITHMS The algorithms are defined in FCS_COP.1/Mobile_network Cryptographic operation. FCS_CKM.2/Mobile_network Cryptographic key distribution describes how the keys are distributed within the MNO profiles, and FCS_CKM.4/Mobile_network Cryptographic key destruction describes the destruction of the keys.

Platform support functions

O.PPE-PPI All SFRs related to Security Domains (FDP_ACC.1/ISDR, FDP_ACC.1/ECASD, FDP_ACF.1/ISDR and FDP_ACF.1/ECASD) cover this security objective by enforcing a Security Domain access control policy (rules and restrictions) that meets the card content management rules. FMT_MSA.1/PPR and FMT_MSA.1/RAT support these SFRs by ensuring management of the Profile Policy Rules (PPR) and Rules Authorisation Table (RAT) files, which ensure that life-cycle modifications are made according to the authorized policy. FMT_MSA.1/PLATFORM_DATA restricts the state transitions that can apply to Platform data (ISD-P state and Fallback attribute) that are used as security attributes by other security policies of the TSF (ISD-R access control SFP and ISD-P content access control SFP). The objective also requires a secure failure mode as described in FPT_FLS.1/EUICC. FCS_RNG.1 Random number generation is required to support FDP_ACF.1/ECASD. NB: The memory reset is also described as a secure failure mode in FPT_FLS.1/EUICC.

O.eUICC-DOMAIN-RIGHTS The requirements FDP_ACC.1/ISDR Subset access control, FDP_ACF.1/ISDR Security attribute based access control, FDP_ACC.1/ECASD Subset access control, and FDP_ACF.1/ECASD Security attribute based access control ensure that ISD-R and ECASD functionality and content are only accessible to the corresponding authenticated user. FTP_ITC.1/SCP Inter-TSF trusted channel provide the corresponding secure channels to the authorized users. FCS_RNG.1 Random number generation is required to support FDP_ACF.1/ECASD Security attribute based access control.

O.SECURE-CHANNELS All SFRs relative to the ES6 and ES8+ interfaces (FTP_ITC.1/SCP Inter-TSF trusted channel, FPT_TDC.1/SCP Inter-TSF basic TSF data consistency, FDP_UCT.1/SCP Basic data exchange confidentiality, FDP_UIT.1/SCP Data exchange integrity, FDP_ITC.2/SCP Import of user data with security attributes, FCS_CKM.1/SCP-SM Cryptographic key generation, FCS_CKM.2/SCP-MNO Cryptographic key distribution, FDP_IFC.1/SCP Subset information flow control, FDP_IFF.1/SCP Simple security attributes, FCS_CKM.4/SCP-SM Cryptographic key destruction, FCS_CKM.4/SCP-MNO Cryptographic key destruction, FCS_CKM.4/SCP-SM Cryptographic key destruction, FCS_CKM.4/SCP-MNO Cryptographic key destruction) cover this security objective by enforcing Secure Channel Protocol information flow control SFP that ensures that transmitted commands and data are protected from unauthorized disclosure and modification.

Identification and authentication SFRs (FIA_UAU.4/EXT Single-use authentication mechanisms, FIA_ATD.1/User User attribute definition, FIA_UID.1/MNO-SD Timing of identification, FIA_UID.1/EXT Timing of identification FIA_USB.1/MNO-SD User-subject binding, FIA_USB.1/EXT, FIA_UAU.1/EXT Timing of authentication) support this security objective by requiring authentication and identification from the distant SM-DP+ and MNO OTA Platform in order to establish these secure channels.

FIA_ATD.1/User User attribute definition, FMT_MSA.1/CERT_KEYS Management of security attributes and FMT_MSA.3/Static attribute initialisation address the management of the security attributes used by the SFP.

FMT_SMF.1/EUICC and FMT_SMR.1/EUICC support these SFRs by providing management of roles and management of functions.

O.INTERNAL-SECURE-CHANNELS FPT_EMS.1/TOE Emanation ensures that secret data stored or transmitted within the TOE shall not be disclosed in cases of side channel attacks. This includes in particular the shared secrets transmitted between ECASD and ISD-R/ISD-P. FDP_SDI.1/Stored data integrity monitoring ensures that the shared secret cannot be modified during this transmission.

FDP_RIP.1/EUICC /Subset residual information protection ensures that the shared secret cannot be recovered from deallocated resources.

8.3.2 Rationale tables of Security Objectives and SFRs

Security Objectives	Security Functional Requirements	Rationale
O.SID	FIA_ATD.1/AID, FIA_UID.2/AID, FMT_MSA.1/JCRE, FMT_MSA.1/ADEL, FMT_MSA.3/ADEL, FMT_MSA.3/FIREWALL, FMT_MSA.1/CM, FMT_MSA.3/CM, FDP_ITC.2/Installer, FMT_SMF.1/CM, FMT_SMF.1/ADEL, FMT_MTD.1/JCRE, FMT_MTD.3/JCRE, FIA_USB.1/AID, FMT_MSA.1/JCVM, FMT_MSA.3/JCVM	Section 8.3.1
O.FIREWALL	FDP_IFC.1/JCVM, FDP_IFF.1/JCVM, FMT_SMR.1/Installer, FMT_MSA.1/CM, FMT_MSA.3/CM, FMT_SMR.1/CM, FMT_MSA.3/FIREWALL, FMT_SMR.1/Firewall, FMT_MSA.1/ADEL, FMT_MSA.3/ADEL, FMT_SMR.1/ADEL, FMT_MSA.1/JCRE, FDP_ITC.2/Installer, FDP_ACC.2/FIREWALL, FDP_ACF.1/FIREWALL, FMT_SMF.1/ADEL, FMT_SMF.1/CM, FMT_SMF.1/Firewall, FMT_MSA.2/FIREWALL_JCVM, FMT_MTD.1/JCRE, FMT_MTD.3/JCRE, FMT_MSA.1/JCVM, FMT_MSA.3/JCVM, FDP_ACC.2/RV_Stack, FDP_ACF.1/RV_Stack, FMT_MSA.1/RV_Stack, FMT_MSA.2/RV_Stack, FMT_MSA.3/RV_Stack, FMT_SMF.1/RV_Stack, FDP_ACC.2/RV_Heap, FDP_ACF.1/RV_Heap, FMT_MSA.1/RV_Heap, FMT_MSA.2/RV_Heap, FMT_MSA.3/RV_Heap, FMT_SMF.1/RV_Heap, FDP_ACC.2/RV_Transient, FDP_ACF.1/RV_Transient, FMT_MSA.1/RV_Transient, FMT_MSA.2/RV_Transient, FMT_MSA.3/RV_Transient, FMT_SMF.1/RV_Transient	Section 8.3.1
O.GLOBAL_ARRAYS_CONFID	FDP_IFC.1/JCVM, FDP_IFF.1/JCVM, FDP_RIP.1/bArray, FDP_RIP.1/APDU, FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL, FDP_RIP.1/TRANSIENT, FDP_RIP.1/GlobalArray	Section 8.3.1
O.GLOBAL_ARRAYS_INTEG	FDP_IFC.1/JCVM, FDP_IFF.1/JCVM	Section 8.3.1
O.NATIVE	FDP_ACF.1/FIREWALL	Section 8.3.1
O.REALLOCATION	FDP_RIP.1/ABORT, FDP_RIP.1/APDU, FDP_RIP.1/bArray, FDP_RIP.1/KEYS, FDP_RIP.1/TRANSIENT, FDP_RIP.1/OBJECTS, FDP_RIP.1/ADEL, FDP_RIP.1/ODEL, FDP_RIP.1/GlobalArray	Section 8.3.1

Security Objectives	Security Functional Requirements	Rationale
O.RESOURCES	FAU_ARP.1, FDP_ROL.1/FIREWALL, FMT_SMR.1/Installer, FMT_SMR.1/Firewall, FMT_SMR.1/ADEL, FPT_FLS.1/Installer, FPT_FLS.1/ODEL, FPT_FLS.1/VM, FPT_FLS.1/ADEL, FPT_RCV.3/Installer, FMT_SMR.1/CM, FMT_SMF.1/ADEL, FMT_SMF.1/CM, FMT_SMF.1/Firewall, FMT_MTD.1/JCRE, FMT_MTD.3/JCRE	Section 8.3.1
O.ARRAY_VIEWS_CONFID	FDP_IFC.1/JCVM, FDP_IFF.1/JCVM	Section 8.3.1
O.ARRAY_VIEWS_INTEG	FDP_IFC.1/JCVM, FDP_IFF.1/JCVM	Section 8.3.1
O.ALARM	FPT_FLS.1/Installer, FPT_FLS.1/VM, FPT_FLS.1/ADEL, FPT_FLS.1/ODEL, FAU_ARP.1	Section 8.3.1
O.CIPHER	FCS_CKM.1/CM-SCP, FCS_CKM.4/CM-SCP, FCS_COP.1/Disp, FPR_UNO.1, FCS_COP.1/Patch, FCS_CKM.4/SUCI, FCS_COP.1/SUCI	Section 8.3.1
O.KEY-MNGT	FCS_CKM.1/CM-SCP, FCS_CKM.4/CM-SCP, FCS_COP.1/Disp, FPR_UNO.1, FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/APDU, FDP_RIP.1/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/TRANSIENT, FDP_RIP.1/ADEL, FDP_SDI.2/DATA, FCS_COP.1/Patch, FCS_CKM.4/SUCI, FDP_RIP.1/GlobalArray	Section 8.3.1
O.PIN-MNGT	FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/APDU, FDP_RIP.1/bArray, FDP_RIP.1/KEYS, FDP_RIP.1/ABORT, FDP_RIP.1/TRANSIENT, FPR_UNO.1, FDP_RIP.1/ADEL, FDP_ROL.1/FIREWALL, FDP_SDI.2/DATA, FDP_ACC.2/FIREWALL, FDP_ACF.1/FIREWALL, FDP_RIP.1/GlobalArray	Section 8.3.1
O.TRANSACTION	FDP_ROL.1/FIREWALL, FDP_RIP.1/ABORT, FDP_RIP.1/APDU, FDP_RIP.1/bArray, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/TRANSIENT, FDP_RIP.1/ODEL, FDP_RIP.1/GlobalArray	Section 8.3.1
O.RNG	FCS_RNG.1	Section 8.3.1
O.OBJ-DELETION	FDP_RIP.1/ODEL, FPT_FLS.1/ODEL	Section 8.3.1
O.DELETION	FDP_ACC.2/ADEL, FDP_ACF.1/ADEL, FDP_RIP.1/ADEL, FMT_MSA.1/ADEL, FMT_MSA.3/ADEL, FPT_FLS.1/ADEL, FMT_SMR.1/ADEL, FPT_RCV.3/Installer	Section 8.3.1
O.LOAD	FCO_NRO.2/CM, FDP_IFC.2/CM, FDP_IFF.1/CM, FDP_UIT.1/CM, FIA_UID.1/CM, FTP_ITC.1/CM	Section 8.3.1

Security Objectives	Security Functional Requirements	Rationale
O.INSTALL	FDP_ITC.2/Installer, FPT_FLS.1/Installer, FPT_RCV.3/Installer	Section 8.3.1
O.SCP.SUPPORT	FPT_RCV.4/SCP, FCS_CKM.1/CM-SCP, FCS_COP.1/Disp, FCS_COP.1/CM-SCP, FCS_COP.1/Patch, FCS_CKM.4/CM-SCP	Section 8.3.1
O.SCP.IC	FCS_RNG.1, FPT_EMS.1, FPR_UNO.1	Section 8.3.1
O.SCP.RECOVERY	FPT_RCV.4/SCP, FAU_ARP.1, FPT_FLS.1/VM	Section 8.3.1
O.CARD_MANAGEMENT	FDP_ACC.2/ADEL, FDP_ACF.1/ADEL, FDP_RIP.1/ADEL, FMT_MSA.1/ADEL, FMT_MSA.3/ADEL, FMT_SMR.1/ADEL, FPT_FLS.1/ADEL, FDP_ITC.2/Installer, FPT_FLS.1/Installer, FPT_RCV.3/Installer, FDP_UIT.1/CM, FDP_ROL.1/FIREWALL, FPT_FLS.1/ODEL, FPT_TST.1, FIA_UID.1/CM, FDP_IFF.1/CM, FMT_MSA.1/CM, FMT_MSA.3/CM, FTP_ITC.1/CM, FDP_IFC.2/CM, FCO_NRO.2/CM_DAP, FIA_UAU.4/CardIssuer, FPT_TDC.1/CM, FMT_SMF.1/CM, FMT_SMR.1/CM, FIA_UAU.1/CM, FPR_UNO.1	Section 8.3.1
O.PATCH_LOADING	FDP_ACC.2/Patch, FDP_ACF.1/Patch, FCS_COP.1/Patch, FTP_ITC.1/CM, FCS_COP.1/CM-SCP, FDP_UIT.1/Patch, FDP_ITC.1/Patch, FDP_UCT.1/Patch, FAU_STG.2/Patch, FPT_TST.1	Section 8.3.1
O.SENSITIVE_ARRAYS_INTEG	FDP_SDI.2/ARRAY	Section 8.3.1
O.PROOF_OF_IDENTITY	FIA_API.1, FCS_CKM.1/SUCI	Section 8.3.1
O.OPERATE	FPT_FLS.1/EUICC, FPT_TDC.1/VM	Section 8.3.1
O.API	FDP_IFC.1/Platform_services, FDP_IFF.1/Platform_services, FPT_FLS.1/Platform_services, FMT_SMR.1/EUICC, FMT_SMF.1/EUICC, FMT_MSA.3	Section 8.3.1
O.DATA-CONFIDENTIALITY	FDP_RIP.1/EUICC, FDP_UCT.1/SCP, FDP_ACC.1/ECASD, FDP_ACC.1/ISDR, FCS_COP.1/Mobile_network, FCS_CKM.4/Mobile_network, FCS_CKM.2/Mobile_network, FPT_EMS.1	Section 8.3.1
O.DATA-INTEGRITY	FDP_UIT.1/SCP, FDP_ACC.1/ISDR, FDP_ACC.1/ECASD, FDP_SDI.1, FPT_TST.1	Section 8.3.1
O.ALGORITHMS	FCS_COP.1/Mobile_network, FCS_CKM.2/Mobile_network, FCS_CKM.4/Mobile_network	Section 8.3.1
O.PPE-PPI	FMT_MSA.1/PLATFORM_DATA, FMT_MSA.1/PPR, FMT_MSA.1/RAT,	Section 8.3.1

Security Objectives	Security Functional Requirements	Rationale
	FPT_FLS.1/EUICC, FDP_ACC.1/ISDR, FDP_ACF.1/ISDR, FDP_ACC.1/ECASD, FDP_ACF.1/ECASD, FCS_RNG.1	
O.eUICC-DOMAIN-RIGHTS	FDP_ACF.1/ISDR, FDP_ACC.1/ISDR, FDP_ACC.1/ECASD, FDP_ACF.1/ECASD, FTP_ITC.1/SCP, FCS_RNG.1	Section 8.3.1
O.SECURE-CHANNELS	FTP_ITC.1/SCP, FPT_TDC.1/SCP, FDP_UCT.1/SCP, FDP_UIT.1/SCP, FDP_ITC.2/SCP, FCS_CKM.1/SCP-SM, FCS_CKM.2/SCP-MNO, FIA_UAU.4/EXT, FIA_ATD.1/User, FMT_MSA.1/CERT_KEYS, FMT_MSA.3, FDP_IFC.1/SCP, FDP_IFF.1/SCP, FIA_UID.1/MNO-SD, FCS_CKM.4/SCP-SM, FCS_CKM.4/SCP-MNO, FIA_USB.1/MNO-SD, FIA_USB.1/EXT, FMT_SMF.1/EUICC, FMT_SMR.1/EUICC, FIA_UAU.1/EXT, FIA_UID.1/EXT	Section 8.3.1
O.INTERNAL-SECURE-CHANNELS	FDP_RIP.1/EUICC, FDP_SDI.1, FPT_EMS.1	Section 8.3.1

Table 12 Security Objectives and SFRs - Coverage

Security Functional Requirements	Security Objectives	Rationale
FDP_ACC.2/FIREWALL	O.FIREWALL, O.PIN-MNGT	
FDP_ACF.1/FIREWALL	O.FIREWALL, O.NATIVE, O.PIN-MNGT	
FDP_IFC.1/JCVM	O.FIREWALL, O.GLOBAL_ARRAYS_CONFID, O.GLOBAL_ARRAYS_INTEG, O.ARRAY_VIEWS_CONFID, O.ARRAY_VIEWS_INTEG	
FDP_IFF.1/JCVM	O.FIREWALL, O.GLOBAL_ARRAYS_CONFID, O.GLOBAL_ARRAYS_INTEG, O.ARRAY_VIEWS_CONFID, O.ARRAY_VIEWS_INTEG	
FDP_RIP.1/OBJECTS	O.GLOBAL_ARRAYS_CONFID, O.REALLOCATION, O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION	
FMT_MSA.1/JCRE	O.SID, O.FIREWALL	
FMT_MSA.1/JCVM	O.SID, O.FIREWALL	
FMT_MSA.2/FIREWALL_JCVM	O.FIREWALL	
FMT_MSA.3/FIREWALL	O.SID, O.FIREWALL	
FMT_MSA.3/JCVM	O.SID, O.FIREWALL	
FMT_SMF.1/Firewall	O.FIREWALL, O.RESOURCES	
FMT_SMR.1/Firewall	O.FIREWALL, O.RESOURCES	
FDP_RIP.1/ABORT	O.GLOBAL_ARRAYS_CONFID, O.REALLOCATION, O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION	

Security Functional Requirements	Security Objectives	Rationale
FDP_RIP.1/APDU	O.GLOBAL_ARRAYS_CONFID, O.REALLOCATION, O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION	
FDP_RIP.1/bArray	O.GLOBAL_ARRAYS_CONFID, O.REALLOCATION, O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION	
FDP_RIP.1/KEYS	O.GLOBAL_ARRAYS_CONFID, O.REALLOCATION, O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION	
FDP_RIP.1/TRANSIENT	O.GLOBAL_ARRAYS_CONFID, O.REALLOCATION, O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION	
FDP_ROL.1/FIREWALL	O.RESOURCES, O.PIN-MNGT, O.TRANSACTION, O.CARD_MANAGEMENT	
FDP_RIP.1/GlobalArray	O.GLOBAL_ARRAYS_CONFID, O.REALLOCATION, O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION	
FCS_CKM.1/CM-SCP	O.CIPHER, O.KEY-MNGT, O.SCP.SUPPORT	
FCS_CKM.4/CM-SCP	O.CIPHER, O.KEY-MNGT, O.SCP.SUPPORT	
FCS_COP.1/Disp	O.CIPHER, O.KEY-MNGT, O.SCP.SUPPORT	
FAU_ARP.1	O.RESOURCES, O.ALARM, O.SCP.RECOVERY	
FDP_SDI.2/DATA	O.KEY-MNGT, O.PIN-MNGT	
FPR_UNO.1	O.CIPHER, O.KEY-MNGT, O.PIN-MNGT, O.SCP.IC, O.CARD_MANAGEMENT	
FPT_FLS.1/VM	O.RESOURCES, O.ALARM, O.SCP.RECOVERY	
FPT_TDC.1/VM	O.OPERATE	
FIA_ATD.1/AID	O.SID	
FIA_UID.2/AID	O.SID	
FIA_USB.1/AID	O.SID	
FMT_MTD.1/JCRE	O.SID, O.FIREWALL, O.RESOURCES	
FMT_MTD.3/JCRE	O.SID, O.FIREWALL, O.RESOURCES	
FDP_ITC.2/Installer	O.SID, O.FIREWALL, O.INSTALL, O.CARD_MANAGEMENT	
FMT_SMR.1/Installer	O.FIREWALL, O.RESOURCES	
FPT_FLS.1/Installer	O.RESOURCES, O.ALARM, O.INSTALL, O.CARD_MANAGEMENT	
FPT_RCV.3/Installer	O.RESOURCES, O.DELETION, O.INSTALL, O.CARD_MANAGEMENT	
FDP_ACC.2/ADEL	O.DELETION, O.CARD_MANAGEMENT	
FDP_ACF.1/ADEL	O.DELETION, O.CARD_MANAGEMENT	
FDP_RIP.1/ADEL	O.GLOBAL_ARRAYS_CONFID, O.REALLOCATION, O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION, O.DELETION, O.CARD_MANAGEMENT	

Security Functional Requirements	Security Objectives	Rationale
FMT_MSA.1/ADEL	O.SID, O.FIREWALL, O.DELETION, O.CARD_MANAGEMENT	
FMT_MSA.3/ADEL	O.SID, O.FIREWALL, O.DELETION, O.CARD_MANAGEMENT	
FMT_SMF.1/ADEL	O.SID, O.FIREWALL, O.RESOURCES	
FMT_SMR.1/ADEL	O.FIREWALL, O.RESOURCES, O.DELETION, O.CARD_MANAGEMENT	
FPT_FLS.1/ADEL	O.RESOURCES, O.ALARM, O.DELETION, O.CARD_MANAGEMENT	
FDP_RIP.1/ODEL	O.GLOBAL_ARRAYS_CONFID, O.REALLOCATION, O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION, O.OBJ-DELETION	
FPT_FLS.1/ODEL	O.RESOURCES, O.ALARM, O.OBJ-DELETION, O.CARD_MANAGEMENT	
FCO_NRO.2/CM	O.LOAD	
FDP_IFC.2/CM	O.LOAD, O.CARD_MANAGEMENT	
FDP_IFF.1/CM	O.LOAD, O.CARD_MANAGEMENT	
FDP_UIT.1/CM	O.LOAD, O.CARD_MANAGEMENT	
FIA_UID.1/CM	O.LOAD, O.CARD_MANAGEMENT	
FMT_MSA.1/CM	O.SID, O.FIREWALL, O.CARD_MANAGEMENT	
FMT_MSA.3/CM	O.SID, O.FIREWALL, O.CARD_MANAGEMENT	
FMT_SMF.1/CM	O.SID, O.FIREWALL, O.RESOURCES, O.CARD_MANAGEMENT	
FMT_SMR.1/CM	O.FIREWALL, O.RESOURCES, O.CARD_MANAGEMENT	
FTP_ITC.1/CM	O.LOAD, O.CARD_MANAGEMENT, O.PATCH_LOADING	
FPT_TST.1	O.CARD_MANAGEMENT, O.PATCH_LOADING, O.DATA-INTEGRITY	
FCO_NRO.2/CM_DAP	O.CARD_MANAGEMENT	
FIA_UAU.1/CM	O.CARD_MANAGEMENT	
FIA_UAU.4/CardIssuer	O.CARD_MANAGEMENT	
FPT_TDC.1/CM	O.CARD_MANAGEMENT	
FCS_COP.1/CM-SCP	O.SCP.SUPPORT, O.PATCH_LOADING	
FDP_ACC.2/Patch	O.PATCH_LOADING	
FDP_ACF.1/Patch	O.PATCH_LOADING	
FDP_UCT.1/Patch	O.PATCH_LOADING	
FDP_ITC.1/Patch	O.PATCH_LOADING	

Security Functional Requirements	Security Objectives	Rationale
FCS_COP.1/Patch	O.CIPHER, O.KEY-MNGT, O.SCP.SUPPORT, O.PATCH_LOADING	
FDP_UIT.1/Patch	O.PATCH_LOADING	
FAU_STG.2/Patch	O.PATCH_LOADING	
FPT_RCV.4/SCP	O.SCP.SUPPORT, O.SCP.RECOVERY	
FCS_RNG.1	O.RNG, O.SCP.IC, O.PPE-PPI, O.eUICC-DOMAIN-RIGHTS	
FDP_ACC.2/RV_Stack	O.FIREWALL	
FDP_ACF.1/RV_Stack	O.FIREWALL	
FMT_MSA.1/RV_Stack	O.FIREWALL	
FMT_MSA.2/RV_Stack	O.FIREWALL	
FMT_MSA.3/RV_Stack	O.FIREWALL	
FMT_SMF.1/RV_Stack	O.FIREWALL	
FDP_ACC.2/RV_Heap	O.FIREWALL	
FDP_ACF.1/RV_Heap	O.FIREWALL	
FMT_MSA.1/RV_Heap	O.FIREWALL	
FMT_MSA.2/RV_Heap	O.FIREWALL	
FMT_MSA.3/RV_Heap	O.FIREWALL	
FMT_SMF.1/RV_Heap	O.FIREWALL	
FDP_ACC.2/RV_Transient	O.FIREWALL	
FDP_ACF.1/RV_Transient	O.FIREWALL	
FMT_MSA.1/RV_Transient	O.FIREWALL	
FMT_MSA.2/RV_Transient	O.FIREWALL	
FMT_MSA.3/RV_Transient	O.FIREWALL	
FMT_SMF.1/RV_Transient	O.FIREWALL	
FDP_SDI.2/ARRAY	O.SENSITIVE_ARRAYS_INTEG	
FIA_UAU.1/EXT	O.SECURE-CHANNELS	
FIA_USB.1/EXT	O.SECURE-CHANNELS	
FIA_UAU.4/EXT	O.SECURE-CHANNELS	
FIA_UID.1/MNO-SD	O.SECURE-CHANNELS	
FIA_USB.1/MNO-SD	O.SECURE-CHANNELS	
FIA_API.1	O.PROOF_OF_IDENTITY	
FIA_UID.1/EXT	O.SECURE-CHANNELS	
FIA_ATD.1/User	O.SECURE-CHANNELS	
FDP_IFC.1/SCP	O.SECURE-CHANNELS	

Security Functional Requirements	Security Objectives	Rationale
FDP_IFF.1/SCP	O.SECURE-CHANNELS	
FTP_ITC.1/SCP	O.eUICC-DOMAIN-RIGHTS, O.SECURE-CHANNELS	
FDP_ITC.2/SCP	O.SECURE-CHANNELS	
FPT_TDC.1/SCP	O.SECURE-CHANNELS	
FDP_UCT.1/SCP	O.DATA-CONFIDENTIALITY, O.SECURE-CHANNELS	
FDP_UIT.1/SCP	O.DATA-INTEGRITY, O.SECURE-CHANNELS	
FCS_CKM.1/SCP-SM	O.SECURE-CHANNELS	
FCS_CKM.2/SCP-MNO	O.SECURE-CHANNELS	
FCS_CKM.4/SCP-SM	O.SECURE-CHANNELS	
FCS_CKM.4/SCP-MNO	O.SECURE-CHANNELS	
FDP_ACC.1/ISDR	O.DATA-CONFIDENTIALITY, O.DATA-INTEGRITY, O.PPE-PPI, O.eUICC-DOMAIN-RIGHTS	
FDP_ACF.1/ISDR	O.PPE-PPI, O.eUICC-DOMAIN-RIGHTS	
FDP_ACC.1/ECASD	O.DATA-CONFIDENTIALITY, O.DATA-INTEGRITY, O.PPE-PPI, O.eUICC-DOMAIN-RIGHTS	
FDP_ACF.1/ECASD	O.PPE-PPI, O.eUICC-DOMAIN-RIGHTS	
FDP_IFC.1/Platform_services	O.API	
FDP_IFF.1/Platform_services	O.API	
FPT_FLS.1/Platform_services	O.API	
FPT_EMS.1	O.SCP.IC, O.DATA-CONFIDENTIALITY, O.INTERNAL-SECURE-CHANNELS	
FDP_SDI.1	O.DATA-INTEGRITY, O.INTERNAL-SECURE-CHANNELS	
FDP_RIP.1/EUICC	O.DATA-CONFIDENTIALITY, O.INTERNAL-SECURE-CHANNELS	
FPT_FLS.1/EUICC	O.OPERATE, O.PPE-PPI	
FMT_MSA.1/PLATFORM_DATA	O.PPE-PPI	
FMT_MSA.1/PPR	O.PPE-PPI	
FMT_MSA.1/CERT_KEYS	O.SECURE-CHANNELS	
FMT_SMF.1/EUICC	O.API, O.SECURE-CHANNELS	
FMT_SMR.1/EUICC	O.API, O.SECURE-CHANNELS	
FMT_MSA.1/RAT	O.PPE-PPI	
FMT_MSA.3	O.API, O.SECURE-CHANNELS	
FCS_COP.1/Mobile_network	O.DATA-CONFIDENTIALITY, O.ALGORITHMS	
FCS_CKM.2/Mobile_network	O.DATA-CONFIDENTIALITY, O.ALGORITHMS	
FCS_CKM.4/Mobile_network	O.DATA-CONFIDENTIALITY, O.ALGORITHMS	

Security Functional Requirements	Security Objectives	Rationale
FCS_CKM.1/SUCI	O.PROOF_OF_IDENTITY	
FCS_CKM.4/SUCI	O.CIPHER, O.KEY-MNGT	
FCS_COP.1/SUCI	O.CIPHER	

Table 13 SFRs and Security Objectives

8.3.3 Dependencies

8.3.3.1 SFRs Dependencies

Requirements	CC Dependencies	Satisfied Dependencies
FDP_ITC.2/Installer	(FDP_ACC.1 or FDP_IFC.1) and (FPT_TDC.1) and (FTP_ITC.1 or FTP_TRP.1)	FPT_TDC.1/VM, FDP_IFC.2/CM, FTP_ITC.1/CM
FMT_SMR.1/Installer	(FIA_UID.1)	
FPT_FLS.1/Installer	No Dependencies	
FPT_RCV.3/Installer	(AGD_OPE.1)	AGD_OPE.1
FDP_ACC.2/ADEL	(FDP_ACF.1)	FDP_ACF.1/ADEL
FDP_ACF.1/ADEL	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.2/ADEL, FMT_MSA.3/ADEL
FDP_RIP.1/ADEL	No Dependencies	
FMT_MSA.1/ADEL	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.2/ADEL, FMT_SMF.1/ADEL, FMT_SMR.1/ADEL
FMT_MSA.3/ADEL	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/ADEL, FMT_SMR.1/ADEL
FMT_SMF.1/ADEL	No Dependencies	
FMT_SMR.1/ADEL	(FIA_UID.1)	
FPT_FLS.1/ADEL	No Dependencies	
FDP_RIP.1/ODEL	No Dependencies	
FPT_FLS.1/ODEL	No Dependencies	
FDP_SDI.2/ARRAY	No Dependencies	
FIA_UAU.1/EXT	(FIA_UID.1)	FIA_UID.1/MNO-SD, FIA_UID.2/AID
FIA_USB.1/EXT	(FIA_ATD.1)	FIA_ATD.1/User
FIA_UAU.4/EXT	No Dependencies	
FIA_UID.1/MNO-SD	No Dependencies	
FIA_USB.1/MNO-SD	(FIA_ATD.1)	FIA_ATD.1/User

Requirements	CC Dependencies	Satisfied Dependencies
FIA_API.1	No Dependencies	
FIA_UID.1/EXT	No Dependencies	
FIA_ATD.1/User	No Dependencies	
FDP_IFC.1/SCP	(FDP_IFF.1)	FDP_IFF.1/SCP
FDP_IFF.1/SCP	(FDP_IFC.1) and (FMT_MSA.3)	FDP_IFC.1/SCP, FMT_MSA.3
FTP_ITC.1/SCP	No Dependencies	
FDP_ITC.2/SCP	(FDP_ACC.1 or FDP_IFC.1) and (FPT_TDC.1) and (FTP_ITC.1 or FTP_TRP.1)	FDP_IFC.1/SCP, FTP_ITC.1/SCP, FPT_TDC.1/SCP
FPT_TDC.1/SCP	No Dependencies	
FDP_UCT.1/SCP	(FDP_ACC.1 or FDP_IFC.1) and (FTP_ITC.1 or FTP_TRP.1)	FDP_IFC.1/SCP, FTP_ITC.1/SCP
FDP_UIT.1/SCP	(FDP_ACC.1 or FDP_IFC.1) and (FTP_ITC.1 or FTP_TRP.1)	FDP_IFC.1/SCP, FTP_ITC.1/SCP
FCS_CKM.1/SCP-SM	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_CKM.4/SCP-SM, FCS_COP.1/Disp, FCS_COP.1/CM-SCP
FCS_CKM.2/SCP-MNO	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FDP_ITC.2/SCP, FCS_CKM.4/SCP-MNO
FCS_CKM.4/SCP-SM	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2)	FDP_ITC.2/SCP, FCS_CKM.1/SCP-SM
FCS_CKM.4/SCP-MNO	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2)	FDP_ITC.2/SCP, FCS_CKM.1/SCP-SM
FDP_ACC.1/ISDR	(FDP_ACF.1)	FDP_ACF.1/ISDR
FDP_ACF.1/ISDR	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.1/ISDR, FMT_MSA.3
FDP_ACC.1/ECASD	(FDP_ACF.1)	FDP_ACF.1/ECASD
FDP_ACF.1/ECASD	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.1/ECASD, FMT_MSA.3
FDP_IFC.1/Platform_services	(FDP_IFF.1)	FDP_IFF.1/Platform_services
FDP_IFF.1/Platform_services	(FDP_IFC.1) and (FMT_MSA.3)	FDP_IFC.1/Platform_services, FMT_MSA.3
FPT_FLS.1/Platform_services	No Dependencies	
FPT_EMS.1	No Dependencies	
FDP_SDI.1	No Dependencies	

Requirements	CC Dependencies	Satisfied Dependencies
FDP_RIP.1/EUICC	No Dependencies	
FPT_FLS.1/EUICC	No Dependencies	
FMT_MSA.1/PLATFORM_DATA	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.1/ISDR, FMT_SMF.1/EUICC, FMT_SMR.1/EUICC
FMT_MSA.1/PPR	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.1/ISDR, FMT_SMF.1/EUICC, FMT_SMR.1/EUICC
FMT_MSA.1/CERT_KEYS	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.1/ISDR, FMT_SMF.1/EUICC, FMT_SMR.1/EUICC
FMT_SMF.1/EUICC	No Dependencies	
FMT_SMR.1/EUICC	(FIA_UID.1)	FIA_UID.1/MNO-SD
FMT_MSA.1/RAT	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.1/ISDR, FMT_SMF.1/EUICC, FMT_SMR.1/EUICC
FMT_MSA.3	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/PLATFORM_DATA, FMT_MSA.1/PPR, FMT_MSA.1/CERT_KEYS, FMT_SMR.1/EUICC, FMT_MSA.1/RAT
FCS_COP.1/Mobile_network	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FDP_ITC.2/SCP, FCS_CKM.4/Mobile_network
FCS_CKM.2/Mobile_network	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FDP_ITC.2/SCP, FCS_CKM.4/SCP-MNO
FCS_CKM.4/Mobile_network	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2)	FDP_ITC.2/SCP
FCS_CKM.1/SUCI	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_CKM.4/SUCI, FCS_COP.1/SUCI, FCS_COP.1/Disp
FCS_CKM.4/SUCI	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2)	FCS_CKM.1/CM-SCP
FCS_COP.1/SUCI	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/SUCI, FCS_CKM.4/SUCI, FCS_CKM.1/CM-SCP
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

Requirements	CC Dependencies	Satisfied Dependencies
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/Firewall
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/Firewall, FMT_SMR.1/Firewall
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/Firewall
FMT_MSA.3/FIREWALL	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/JCRE, FMT_MSA.1/JCVM, FMT_SMR.1/Firewall
FMT_MSA.3/JCVM	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/JCVM, FMT_SMR.1/Firewall
FMT_SMF.1/Firewall	No Dependencies	
FMT_SMR.1/Firewall	(FIA_UID.1)	FIA_UID.2/AID
FDP_RIP.1/ABORT	No Dependencies	
FDP_RIP.1/APDU	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
FDP_RIP.1/GlobalArray	No Dependencies	
FCS_CKM.1/CM-SCP	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_CKM.4/CM-SCP, FCS_COP.1/Disp
FCS_CKM.4/CM-SCP	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2)	FCS_CKM.1/CM-SCP
FCS_COP.1/Disp	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/CM-SCP, FCS_CKM.4/CM-SCP
FAU_ARP.1	(FAU_SAA.1)	
FDP_SDI.2/DATA	No Dependencies	

Requirements	CC Dependencies	Satisfied Dependencies
FPR_UNO.1	No Dependencies	
FPT_FLS.1/VM	No Dependencies	
FPT_TDC.1/VM	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/Firewall, FMT_SMR.1/Firewall
FMT_MTD.3/JCRE	(FMT_MTD.1)	FMT_MTD.1/JCRE
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	No Dependencies	
FCO_NRO.2/CM_DAP	(FIA_UID.1)	FIA_UID.1/CM
FIA_UAU.1/CM	(FIA_UID.1)	FIA_UID.1/CM
FIA_UAU.4/CardIssuer	No Dependencies	
FPT_TDC.1/CM	No Dependencies	
FCS_COP.1/CM-SCP	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/SCP-SM, FCS_CKM.1/CM-SCP, FCS_CKM.4/CM-SCP
FDP_ACC.2/Patch	(FDP_ACF.1)	FDP_ACF.1/Patch
FDP_ACF.1/Patch	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.2/Patch

Requirements	CC Dependencies	Satisfied Dependencies
FDP_UCT.1/Patch	(FDP_ACC.1 or FDP_IFC.1) and (FTP_ITC.1 or FTP_TRP.1)	FTP_ITC.1/CM, FDP_ACC.2/Patch
FDP_ITC.1/Patch	(FDP_ACC.1 or FDP_IFC.1) and (FMT_MSA.3)	FDP_ACC.2/Patch
FCS_COP.1/Patch	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.4/CM-SCP, FDP_ITC.1/Patch
FDP_UIT.1/Patch	(FDP_ACC.1 or FDP_IFC.1) and (FTP_ITC.1 or FTP_TRP.1)	FTP_ITC.1/CM, FDP_ACC.2/Patch
FAU_STG.2/Patch	(FAU_GEN.1)	
FPT_RCV.4/SCP	No Dependencies	
FCS_RNG.1	No Dependencies	
FDP_ACC.2/RV_Stack	(FDP_ACF.1)	FDP_ACF.1/RV_Stack
FDP_ACF.1/RV_Stack	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.2/RV_Stack, FMT_MSA.3/RV_Stack
FMT_MSA.1/RV_Stack	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FMT_SMR.1/Firewall, FDP_ACC.2/RV_Stack, FMT_SMF.1/RV_Stack
FMT_MSA.2/RV_Stack	(FDP_ACC.1 or FDP_IFC.1) and (FMT_MSA.1) and (FMT_SMR.1)	FMT_SMR.1/Firewall, FDP_ACC.2/RV_Stack, FMT_MSA.1/RV_Stack
FMT_MSA.3/RV_Stack	(FMT_MSA.1) and (FMT_SMR.1)	FMT_SMR.1/Firewall, FMT_MSA.1/RV_Stack
FMT_SMF.1/RV_Stack	No Dependencies	
FDP_ACC.2/RV_Heap	(FDP_ACF.1)	FDP_ACF.1/RV_Heap
FDP_ACF.1/RV_Heap	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.2/RV_Heap, FMT_MSA.3/RV_Heap
FMT_MSA.1/RV_Heap	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FMT_SMR.1/Firewall, FDP_ACC.2/RV_Heap, FMT_SMF.1/RV_Heap
FMT_MSA.2/RV_Heap	(FDP_ACC.1 or FDP_IFC.1) and (FMT_MSA.1) and (FMT_SMR.1)	FMT_SMR.1/Firewall, FDP_ACC.2/RV_Heap, FMT_MSA.1/RV_Heap
FMT_MSA.3/RV_Heap	(FMT_MSA.1) and (FMT_SMR.1)	FMT_SMR.1/Firewall, FMT_MSA.1/RV_Heap

Requirements	CC Dependencies	Satisfied Dependencies
FMT_SMF.1/RV_Heap	No Dependencies	
FDP_ACC.2/RV_Transient	(FDP_ACF.1)	FDP_ACF.1/RV_Transient
FDP_ACF.1/RV_Transient	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.2/RV_Transient, FMT_MSA.3/RV_Transient
FMT_MSA.1/RV_Transient	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FMT_SMR.1/Firewall, FDP_ACC.2/RV_Transient, FMT_SMF.1/RV_Transient
FMT_MSA.2/RV_Transient	(FDP_ACC.1 or FDP_IFC.1) and (FMT_MSA.1) and (FMT_SMR.1)	FMT_SMR.1/Firewall, FDP_ACC.2/RV_Transient, FMT_MSA.1/RV_Transient
FMT_MSA.3/RV_Transient	(FMT_MSA.1) and (FMT_SMR.1)	FMT_SMR.1/Firewall, FMT_MSA.1/RV_Transient
FMT_SMF.1/RV_Transient	No Dependencies	

Table 14 SFRs Dependencies

Rationale for the exclusion of Dependencies

The dependency FIA_UID.1 of FMT_SMR.1/Installer is discarded. 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 discarded. This ST does not require the identification of the "deletion manager" since it can be considered as part of the TSF.

The dependency FMT_SMF.1 of FMT_MSA.1/JCRE is discarded. The dependency between FMT_MSA.1/JCRE and FMT_SMF.1/Firewall is not satisfied because no management functions are required for the Java Card RE.

The dependency FAU_SAA.1 of FAU_ARP.1 is discarded. The dependency of FAU_ARP.1 on FAU_SAA.1 assumes that a "potential security violation" generates an audit event. On the contrary, the events listed in FAU_ARP.1 are self-contained (arithmetic exception, ill-formed bytecodes, access failure) and ask for a straightforward reaction of the TSFs on their occurrence at runtime. The JCVM or other components of the TOE detect these events during their usual working order. Thus, there is no mandatory audit recording in this ST.

The dependency FMT_MSA.3 of FDP_ACF.1/Patch is discarded. The access control TSF according to FDP_ACF.1/Patch uses security attributes that have been defined during personalization, and that are fixed over the whole life time of the TOE. No management of these security attributes (i.e. SFR FMT_MSA.1 and FMT_MSA.3) is necessary here.

The dependency FMT_MSA.3 of FDP_ITC.1/Patch is discarded. The access control TSF according to FDP_ITC.1/Patch uses security attributes that have been defined during personalization, and that are fixed over the whole life time of the TOE. No management of these security attributes (i.e. SFR FMT_MSA.1 and FMT_MSA.3) is necessary here.

The dependency FAU_GEN.1 of FAU_STG.2/Patch is discarded. The FAU_STG.2/Patch is related to the patch. When the identification of the patch is incorrect, the TOE rise a kill Card exception. The

FAU_GEN is then discarded as the card returns only the ATR. There is need to store any audit function.

8.3.3.2 SARs Dependencies

Requirements	CC Dependencies	Satisfied Dependencies
ALC_FLR.1	No 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_IMP.1	(ADV_TDS.3) and (ALC_TAT.1)	ADV_TDS.3, ALC_TAT.1
ADV_TDS.3	(ADV_FSP.4)	ADV_FSP.4
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, 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

Table 15 SARs Dependencies

8.3.4 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.

8.3.5 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.

8.3.6 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.

8.3.7 ALC_FLR.1 Basic flaw remediation

ALC_FLR.1 provides assurance to the users that IDEMIA has policies and procedures to track and correct flaws, and to distribute the flaw information and corrections.

9 TOE Summary Specification

9.1 TOE Summary Specification

9.1.1 eUICC Security Functions

SF_ACCESS

This TSF provides means to control the accesses to the TOE by controlling

- proof of identity, the identity of the eUICC or identities of actors,
- secure distribution, deletion or storage of keyset for Secure Channel Protocol
- the information for plateform services: Data describing the Rules Authorisation Table (RAT) of the eUICC, the rules and parameters between SD and Profile Policy Enabler (PPE), Profile Package Interpreter (PPI) and the Telecom Framework.
- correct operation of the security functions with the failures and consequences on secure initialisation,
- the unauthorized used of commands and authentication to Mobile network: managing the keys securely for Milenage, Tuak with secure distribution, destruction or storage.
- TSF roles and access control policy for ISDR and ECASD.

SF_DATA_PROTECTION

This function ensures that confidentiality and integrity of data are protected. This security function also ensures atomic transactions. Thus secret data stored or transmitted within the TOE are protected in cases of side channel or perturbation attacks. The security function relies also on Javacard and IC protections. The data to be protected can be profiles, data profile, the on-card generated keys or keys distributed along with the Profile, Commands received from SM-DP+ and MNO OTA Platform, PPR received from the MNO OTA Platform, the on-card representative of the SM-DP+: ISP-P, security attributes for access control, output data of ECASD functions, certificates, shared secret internal secure channel, secret data stored or transmitted within the TOE, management data (platform, device RAT),Identity management data. Also the data concerns the Mobile_network crytography for keys and data of Milenage and Tuak, as for SUCI applet. The functions ensures that any previous information content of a resource is made unavailable.

SF_SECURE_DOMAIN

This functions ensures the secure communication and security domains distribution. This security function relies on communication protection measures provided by the Runtime Environment (Javacard services). The eUICC maintains secure channel between ISD and MNO-SD, restricts the modification of Security Domains.This function performs ISD-R and ECASD management with installation, provisioning and credentials management. This function also provide the plateform management for application and profiles. The profile with ISD-P management concerns its installation, loading, enabling and deletion.

9.1.2 Runtime environment Security Functions

SF_ATOMIC_TRANSACTION

This TSF provides means to execute a sequence of modifications and allocations on the persistent memory so that either all of them are completed, or the TOE behaves as if none of them had been attempted. The transaction mechanism is used for updating internal TSF data as well as for performing different functions of the TOE, like installing a new package on the card. This TSF is also available for applet instances through the javacard.framework.JCSystem, javacard.framework.Util and javacardx.framework.util.ArrayLogic classes. The first class provides the applet instances with methods for starting, aborting and committing a sequence of modifications of the persistent memory. The other classes provide methods for atomically copying arrays. This TSF ensures that the following data is never updated conditionally:

- The validated flag of the PINs
- The reason code of the CardException and CardRuntimeException
- Transient objects
- Global arrays, like the APDU buffer and the buffer that the applet instances use to store installation data
- Any intermediate result state in the implementation instance of the Checksum, Signature, Cipher, and Message Digest classes of the JavaCard API.

This TSF is in charge of setting back the state of the persistent memory as it was before they were started, when the following operations specified are not completed:

- Loading and linking of a package
- Installing a new applet instance
- Deleting a package
- Deleting an applet instance
- Collecting unreachable objects
- Reading from and writing to a static field, instance field or array position
- Populating, updating or clearing a cryptographic key
- Modifying a PIN value

Upon deallocation of a resource from any reference to an object instance created during an aborted transaction, any previous information content of the resource is made unavailable.

Finally, this TSF ensures that no transaction is in progress when a method of an applet instance is invoked for installing, deselecting, selecting or processing an APDU sent to the applet instance. Concerning memory limitations on the transaction journal, this TSF guarantees that an exception is thrown when the maximal capacity is reached. The TSF preserves a secure state when such limit is reached. Atomic Transactions are detailed in the chapter Atomicity and Transactions of the [JCRE] and in the documentation associated to the JCSystem class in the [JCAPI].

SF_UNOBSERVABILITY

This function assures that processing based on secure elements of the TOE does not reveal any information on those elements. For example, observation of a PIN verification cannot reveal the PIN value, observation a cryptographic computation cannot give information on the key.

SF_SIGNATURE

This TSF provides the applet instances with a mechanism for generating an electronic signature of a byte array content and verifying an electronic signature contained in a byte array. An electronic signature is made of a hash value of the information to be signed encrypted with a secret key. The verification of the electronic signature includes decrypting the hash value and checking that it actually corresponds to the block of signed bytes.

The signature algorithms are available to the applets through the javacard.Signature class of the Java Card API, ISOSecureMesssaging class and SecureChannel class. The length of the key to be used for the signature is defined by the applet instance when the key is created. Before generating the signature, the TSF verifies that the specified key is suitable for the operation (secret keys for signature generation), that it has been previously initialized, and that is in accordance with the specified signature algorithm (DES, etc). The TSF also checks that it has been provided with all the information necessary for the signature operation. For those algorithms that do not pad the messages, the TSF checks that the information to be signed is block aligned before performing the signature operation. Once the signature operation is performed, the internal TSF data used for the operation like the ICV is cleared. Signature operations are implemented to resist to environmental stress and glitches and include measures for preventing information leakage through covert channels.

Mechanisms of signature for Secure Messaging are available to the applets through the SecureChannel (Global Platform Card 2.3.1" specification). The signature is included in Data Objects.

SF_SECURITY_FUNCTIONS_OF_THE_IC

The TOE uses the security functions of the IC. The list of the security function is presented in the ST lite of the IC component.

SF_RUNTIME_VERIFIER

This security functionality ensures the secure processing of information by ensuring the following elements:

- Stack Control
- Heap Control
- Transient Control

Information on the processing is described on the related FDP_ACF.1.

SF_RANDOM_NUMBER

This TSF provides to card manager, Framework application, applets a mechanism for generating challenges and key values. Random number generators are available to the applets through the RandomData class of the Java Card API. Off-card entity authentication is achieved through the process of initiating a Secure Channel and provides assurance to the card that it is communicating with an authenticated off-card entity. If any step in the off-card authentication process fails, the process shall be restarted (i.e. new session keys generated). The Secure Channel initiation and off-card entity authentication implies the creation of session keys derived from card static key(s).

SF_PATCHING

This function is in charge loading patch code, if needed. The patch contains its identification elements that are used, during audit, to uniquely identify loaded code. The Patch loading, dedicated to update the platform, an applet or a package, can be done from pre-personilization phase to use phase. SF_DATA_INTEGRITY ensure integrity of patch installation.

SF_MESSAGE_DIGEST

This TSF provides the applet instances with a mechanism for generating an (almost) unique value for a byte array content. That value can be used as a short representative of the information contained in the whole byte array. The hashing algorithms are available to the applets through the MessageDigest class of the Java Card API. Before generating the hash value, the TSF verifies that it has been provided with all the information necessary for the hashing operation. For those algorithms that do not pad the messages, the TSF checks that the information is block aligned before computing its hash value.

SF_KEY_MANAGEMENT

This function enables key sets management (PIN). It allows creating updating and deleting key sets. It is used to load keys to the card. It also implements verification of Key sets attributes: key lengths, key types... and enforces the loaded keys integrity

SF_KEY_GENERATION

This TSF enforces the creation and/or the oncard generation of all the cryptographic keys of the card using the method specified in that SFR.

SF_KEY_DESTRUCTION

This TSF disables the use of a key both logically and physically. When a key is cleared, the internal life cycle of the key container is moved to a state in which no operation is allowed. Applet instances may invoke this TSF through the interfaces declared in the javacard.security package of the Java Card API.

SF_KEY_AGREEMENT

This TSF provides the applet instances with a mechanism for supporting key agreement algorithms such as EC Diffie-Hellman [IEEE P1363].

SF_HARDWARE_OPERATING

When needed, at each start up or before first use, a self test of each hardware functional module is done, i.e.: DES, RNG implements a know calculus and checks if the result is correct. When executing, external hardware event can be trigged to prevent attacks or bad use. Temperature, frequency, voltage, light, glitch are considered as abnormal environmental conditions and put the card in frozen state. The TOE shall monitor IC detectors (e.g. out-of-range voltage, temperature, frequency, active shield, memory aging) and shall provide automatic answers to potential security violations through interruption routines that leave the device in a secure state.

The TOE with the IC has detectors of operational conditions. It shall resist to attackers with high-attack potential according to [JIL1] characterisation, in particular, to leakage attacks, intrusive (e.g. probing, fault injection) and non-intrusive (e.g. SPA, DPA, EMA) attacks, operational conditions manipulation (voltage, clock, temperature, etc) and physical attacks aiming at modification of the IC content or behaviour. To be compliant to related SUN Protection Profile [PP0099], the off-card verifier is mandatory in this ST; however, this TOE runs some additional verification at execution time. These verifications ensure that: 1. No read accesses are made to Java Card System code, data belonging to another application, data belonging to the Java Card System, 2. No write accesses are made to another application's code, Java Card System code, another application's data Java Card System or API data, 3. No execution of code is done from a method or from a method fragment belonging to another package (including execution on arbitrary data).

SF_GP_DISPATCHER

While a Security Domain is selected, this function tests for every command, according to the Security Domain life cycle state and the Card life cycle state, if security requirements are needed (if a Secure Channel is required).

SF_FIREWALL

This TSF enforces the Firewall security policy and the information flow control policy at runtime. The former policy controls object sharing between different applet instances, and between applet instances and the Java Card RE. The latter policy controls the access to global data containers shared by all applet instances. This TSF is enforced by the Java Card platform Virtual Machine (Java Card VM). During the execution of an applet, the Java Card VM keeps track of the applet instance that is currently performing an action. This information is known as the currently active context. Two kinds of contexts are considered: applet instances contexts and the Java Card RE context, which has special privileges for accessing objects. The TSF makes no difference between instances of applets defined in the same package: all of them belong to the same active context. On the contrary, instances of applets defined in different packages belong to different contexts. Each object belongs to the context that was active when the object was allocated. Initially, when the Java Card VM is launched, the context corresponding to the applet instance selected for execution becomes the first active context. Each time an instance method is invoked on an object, a context switch is performed, and the owner of the object becomes the new active context. On the contrary, the invocation of a static method does not entail a context switch. Before executing a bytecode that accesses an object, the object's owner is checked against the currently active context in order to determine if access is allowed. Access is determined by the Firewall access control rules specified in the chapter Applet Isolation and Object Sharing of the [JCRE]. Those rules enable controlled sharing of objects through interface methods that the object's owner explicitly exports to other applet instances, and provided that the object's owner explicitly accepts to share it upon request of the method's invoker.

SF_EXCEPTION

In case of abnormal event: data unavailable on an allocation, illegal access to a data, the system owns an internal mechanism that allows to stop the code execution and raise an exception.

SF_ENTITY_AUTHENTICATION/SECURE_CHANNEL

Off-card entity authentication is achieved by initiating a Secure Channel and provides assurance to the card that it is communicating with an authenticated off-card entity. If any step in the off-card authentication process fails, the process shall be restarted (i.e. new session keys generated). The Secure Channel initiation and off-card entity authentication implies the creation of session keys derived from card static key(s).

SF_ENCRYPTION_AND_DECRYPTION

This TSF provides the applet instances with mechanisms for encrypting and decrypting the contents of a byte array.

The ciphering algorithms are available to the applets through the Cipher class of the Java Card API and SecureChannel class. The length of the key to be used for the ciphering operation is defined by the applet instance when the key is generated. Before encrypting or decrypting the byte array, the TSF verifies that the specified key has been previously initialized, and that is in accordance with the specified ciphering algorithm (DES, etc). The TSF also checks that it has been provided with all the information necessary for the encryption/decryption operation. Once the ciphering operation is performed, the internal TSF data used for the operation like the ICV is cleared. Ciphering operations are implemented to resist to environmental stress and glitches and include measures for preventing information leakage through covert channels.

Mechanisms of encrypting and decrypting for Secure Messaging are available to the applets through the SecureChannel (Global Platform Card 2.3.1" specification) and ISOSecureMessaging (Proprietary API **[AGD_PAPI]**) classes.

SF_DATA_INTEGRITY

Some of the data in non volatile memory can be protected. Keys, PIN and patch code are protected with integrity value. When reading and writing operation, the integrity value is checked and maintained valid. In case of incoherency, an exception is raised to prevent the bad use of the data. SecureStore is a mean for protecting JavaCard data in integrity.

SF_DATA_COHERENCY

As coherency of data should be maintained, and as power is provided by the CAD and might be stopped at all moment (by tearing or attacks), a transaction mechanism is provided. When updating data, before writing the new ones, the old ones are saved in a specific memory area. If a failure appears, at the next start-up, if old data are valid in the transaction area, the system restores them for staying in a coherent state.

SF_DAP_VERIFICATION

An Application Provider may require that its Application code to be loaded on the card is checked for integrity and authenticity. The DAP Verification privilege of the Application Provider's Security Domain detailed in Section 9.2.1 of provides this service on behalf of an Application Provider. A Controlling Authority may require that all Application code to be loaded onto the card shall be checked for integrity and authenticity. The Mandated DAP Verification privilege of the Controlling Authority's Security Domain detailed in Section 9.2.1 of provides this service on behalf of the Controlling Authority. The keys and algorithms to be used for DAP Verification or Mandated DAP Verification are implicitly known by the corresponding Security Domain.

SF_CLEARING_OF_SENSITIVE_INFORMATION

This TSF clears all the data containers that hold sensitive information when that information is no longer used or upon the allocation of the resource. This includes:

- The contents of the memory blocks allocated for storing class instances, arrays, static field images and local variables, before allocating a fresh block
- The objects reclaimed by the Java Card VM garbage collector
- The code of the deleted packages
- The objects accessible from a deleted applet instance
- The content of the bArray argument of the Applet.install method after a new applet instance is installed
- The content of CLEAR ON DESELECT transient objects owned by an applet instance that has been deselected when no other applets from the same package are active on the card
- The content of all transient objects after a card reset
- The contents of the cryptographic buffer after performing cryptographic operations
- The Reference to an object instance created during an aborted transaction
- The validated flag of the PINs after a card reset

Application Note:

This function is in charge of clearing the information contained in the objects that are no longer accessible from the installed packages and applet instances. Clearing is performed on demand of an applet instance through the JCSystem.requestObjectDeletion() method.

SF_CARDHOLDER_VERIFICATION

This TSF enables applet instances to authenticate the sender of a request as the true cardholder. Applet instances have access to these services through the OwnerPIN class. Cardholder authentication is performed using the following security attributes:

- A secret enabling to authenticate the cardholder
- The maximum number of consecutive unsuccessful comparison attempts that are admitted
- A counter of the number of consecutive unsuccessful comparison attempts that have been performed so far
- The current life cycle state of the secret (reference value). This state is always updated, even if the modification is in the scope of an open transaction. Each time an attempt is made to compare a value to the reference value, and prior to the comparison being actually performed, if the reference is blocked, then the comparison fails and the reference value is not accessed. Otherwise, the try counter is decremented by one. This operation is always performed, even if it is in the scope of an open transaction. If the comparison is successful, then the try counter is reset to the try limit. When the try counter reaches zero, the reference enters into a blocked state, and cannot be used until it is unblocked. Cardholder Verification Method services are implemented to resist to environmental stress and glitches and include measures for preventing information leakage through covert channels. In particular,

unsuccessful authentication attempts consume the same power and execution time than successful ones. The Cardmanager uses the class OwnerPin to provide the services to the Applet that want benefit of the Shared GP_PIN. The **SF_CARDHOLDER_VERIFICATION implements all Pin verifications: D.PIN, GP.PIN**.

SF_CARD_MANAGEMENT_ENVIRONMENT

This TSF is in charge of initializing and managing the internal data structures of the Card Manager. During the initialization phase of the card, this TSF creates the Installer and the Applet Deletion Manager and initializes their internal data structures. The internal data structures of the Card Manager includes the Package and Applet Registries, which respectively contains the currently loaded packages and the currently installed applet instances, together with their associated AIDs. This TSF is also in charge of dispatching the APDU commands to the applets instances installed on the card and keeping traces of which are the currently active ones. It therefore handles sensitive TSF data of other security functions, like the Firewall.

SF_CARD_CONTENT_MANAGEMENT

This TSF ensures the following functionalities:

- Loading (Section 9.3.5 of [GP2]): This function allows the addition of code to mutable persistent memory in the card. During card content loading, this TSF checks that the required packages are already installed on the card. If one of the required packages does not exist, or if the version installed on the card is not binary compatible with the version required, then the loading of the package is rejected. Loading is also rejected if the version of the CAP format of the package is newer than the one supported by the TOE. If any of those checks fails, a suitable error message is returned to the CAD.
- Installation (Section 9.3.6 of [GP2]): This function allows the Installer to create an instance of a previously loaded Applet subclass and make it selectable. In order to do this, the install() method of the Applet subclass is invoked using the context of that new instance as the currently active context. If this method returns with an exception, the exception is trapped and the smart card rolls back to the state before starting the installation procedure.
- Deletion (Section 9.5 of [GP2]): This function allows the Applet Deletion Manager to remove the code of a package from the card, or to definitely deactivate an applet instance, so that it becomes no longer selectable. This TSF performs physical removal of those packages and applet data stored in NVRAM, while only logical removal is performed for packages in ROM. This TSF checks that the package or applet actually exists, and that no other package or applet depends on it for its execution. In this case, the entry of the package or applet is removed from the registry, and all the objects on which they depend are garbage collected. Otherwise, a suitable error is returned to the CAD. The deletion of the Applet Deletion Manager, the Installer or any of the packages required for implementing the Java Card platform Application Programming Interface (Java Card API) is not allowed.
- Extradition (Section 9.4.1 of [GP2]): This function allows the Installer to associate load files or applet instances to a Security Domain different than their currently associated Security Domain. It is also used to associate a Security Domain to another Security Domain or to itself thus creating Security Domains hierarchies. If this method returns with an exception, the exception is trapped and the smart card rolls back to the state before starting the extradition procedure.
- Registry update (Section 9.4.2 of [GP2]): This function allows the Installer to populate, modify or delete elements of the Registry entry of applet instances. If this method returns with an exception, the exception is trapped and the smart card rolls back to the state before starting the extradition procedure.

9.2 SFRs and TSS

9.2.1 SFRs and TSS - Rationale

Java Card

CoreG_LC Security Functional Requirements

Firewall Policy

FDP_ACC.2/FIREWALL The access control policy is ensured by SF_FIREWALL, it controls whether an instance of an applet class declared in a package (subject) may read, write or execute an instance method (operations) of an object (object).

FDP_ACF.1/FIREWALL FIREWALL Security attribute based access control -which security attributes is attached to which subject/object of the policy- is specified in the SF_FIREWALL.

FDP_IFC.1/JCVM This requirement is fulfilled by SF_FIREWALL, this TSF enforces the information flow control rules of Firewall security policy. It controls whether an applet instance or javacard RE (subject) may store into persistent memory a reference of a global shared data container (objects).

FDP_IFF.1/JCVM This requirement is fulfilled by SF_FIREWALL. This TSF controls operations, based on current active context implemented in SF_FIREWALL.

FDP_RIP.1/OBJECTS SF_CLEARING_OF_SENSITIVE_INFORMATION. The TSF clears the contents of the freshly allocated objects before releasing the object to the applet. On the TSF, memory is cleared when the object is removed during Garbage Collection. All this TSFI lead to Garbage Collection

FMT_MSA.1/JCRE SF_FIREWALL When an instance method is applied to an object, this TSF is in charge of performing a context switch to the context of the object's owner. The TSF is also in charge of dispatching the APDU commands to the applets instances installed on the card and keeping trace of which are the currently active ones.

FMT_MSA.1/JCVM SF_FIREWALL When an instance method is applied to an object, this TSF is in charge of performing a context switch to the context of the object's owner. The TSF is also in charge of dispatching the APDU commands to the applets instances installed on the card and keeping traces of which are the currently active ones.

FMT_MSA.2/FIREWALL_JCVM SF_FIREWALL When an applet instance is selected for execution, this TSF initializes the currently active context with (the context of) that instance. Applet selection includes the verification that the instance actually exists on the card. Then, during the execution of the Java Card VM, this TSF propagates that secure value the other security attributes involved in the Firewall policy (object's owner).

FMT_MSA.3/FIREWALL SF_FIREWALL The TSF initializes the security attributes of the Firewall and Java Card VM security policies when an applet instance is selected for execution, when an instance method is invoked and when an object is allocated. This TSF does not provide means for a subject to override those initial values.

FMT_MSA.3/JCVM SF_FIREWALL. The TSF initializes the security attributes of the Firewall and Java Card VM security policies when an applet instance is selected for execution, when an instance method is invoked and when an object is allocated. This TSF does not provide means for a subject to override those initial values.

FMT_SMF.1/Firewall This SFR is fulfilled by SF_CARD_CONTENT_MANAGEMENT, when an instance method is applied to an object; this TSF is in charge of performing a context switch to the context of the object's owner.

FMT_SMR.1/Firewall This requirement is full filled by SF_FIREWALL, this TSF uses a special value for the currently active context that identifies the Java Card RE (JCRE) and Java Card VM (JCVM).

Application Programming Interface

FDP_RIP.1/ABORT Any reference to an object instance created during an aborted transaction- see SF_ATOMIC_TRANSACTIONS- is cleaned by using SF_CLEARING_OF_SENSITIVE_INFORMATION.

FDP_RIP.1/APDU The TSF SF_CLEARING_OF_SENSITIVE_INFORMATION enforces the clearing of the previous contents of the APDU buffer before processing a new APDU.

FDP_RIP.1/bArray The TSF SF_CLEARING_OF_SENSITIVE_INFORMATION enforces the clearing of the previous contents of the buffer containing the installation data of an applet instance before installing a new one.

FDP_RIP.1/KEYS In order to perform a cryptographic operation, the key involved in the operation has to be copied out of its secure container into the cryptographic buffer of the IC co-processor. This function is in charge of ensuring that such buffer is cleared immediately after completing the operation, the clearing is done by SF_CLEARING_OF_SENSITIVE_INFORMATION.

FDP_RIP.1/TRANSIENT This function is in charge of clearing the information contained in the transient objects when a clearing event arrives (deselection or card reset), invoked by SF_CLEARING_OF_SENSITIVE_INFORMATION.

FDP_ROL.1/FIREWALL SF_ATOMIC_TRANSACTION, when the operations specified are not completed, this TSF is in charge of setting back the state of the persistent memory as it was before they were started. As required in chapter 7 of the [29] and the [32], this TSF does not undo those modifications performed on the RAM, like the modification of the APDU buffer, the installation buffer, the transient objects, the try counters of the PINs and the reason code of the card exceptions. If the commit capacity is reached, this TSF prevents any further modification of the persistent memory.

FDP_RIP.1/GlobalArray The TSF SF_CLEARING_OF_SENSITIVE_INFORMATION enforces the clearing of the previous contents of the buffer containing the installation data of an applet instance before installing a new one.

FCS_CKM.1/CM-SCP This requirement is fulfilled by SF_KEY_GENERATION. It enforces the creation and/or the oncard generation of all the cryptographic keys of the card.

FCS_CKM.4/CM-SCP SF_KEY_DESTRUCTION fulfils this SFR, it enforces the destruction of all the cryptographic keys of the card using the method specified in that SFR.

FCS_COP.1/Disp This SFR is verified by the following set of Security functionalities:

- All signature and verification operation by TDES and AES are fulfilled by SF_SIGNATURE, also fulfilled by SF_KEY_AGREEMENT by providing the applet instances with a mechanism for supporting key agreement algorithms such EC Diffie-Hellman [41].
- This requirement by using SF_ENCRYPTION_AND_DECRYPTION provides the applet instances with a mechanism for encrypting and decrypting the contents of a byte array.
- SF_SIGNATURE permits to hash functions with SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512. It is also fulfilled by SF_MESSAGE_DIGEST by providing applet instances with a mechanism for generating an (almost) unique value for the contents of a byte array. Also fulfilled by SF_KEY_AGREEMENT by providing the applet instances with a mechanism for supporting key agreement algorithms such as EC Diffie-Hellman [41].

Card Security Management

FAU_ARP.1 The SF_FIREWALL throws an instance of the SecurityException class when an attempt to violate a security policy rule is detected.

FDP_SDI.2/DATA The TSF SF_DATA_INTEGRITY ensures integrity of PIN, Keys and application code (package)(CRC 16 and 32). A loss of integrity increases killcard counter.

FPR_UNO.1 The TSF SF_UNOBSERVABILITY ensures no user is able to observe PIN values when authentication of the cardholder.

FPT_FLS.1/VM This SFR is enforced by the following TSF:

- SF_ATOMIC_TRANSACTIONS: card tearing and power failures and abortion of a transaction in an unexpected context
- SF_FIREWALL: violations of the Firewall access control rules,
- SF_CARD_CONTENT_MANAGEMENT: insufficient resources to install a package and CAP file inconsistency errors.
- SF_CLEARING_OF_SENSITIVE_INFORMATION: ensures the erase of previous information stored, like the flags of pin or reason code contained in the CardException or CardRuntimeException.

FPT_TDC.1/VM This SFR is fulfilled by F_CARD_CONTENT_MANAGEMENT_ENVIRONMENT. It interprets cap files: bytes code and data arguments.

AID Management

FIA_ATD.1/AID This SFR is fulfilled by SF_ARD_CONTENT_MANAGEMENT: It controls the addition of new entries in the Applet Registry. Each time a new entry is added, the TSF controls that it contains the information specified in that SFR. This is done on package loading and applet installation.

FIA_UID.2/AID The TSF SF_FIREWALL identifies the applet instance requesting access to objects through the currently active context. Retrieving the currently active context always precedes the execution of the bytecodes under the control of the Firewall, as this information is required for checking the premises of its access control rules.

FIA_USB.1/AID The TSF SF_FIREWALL uses the security attribute introduced in the SFR to check whether an applet instance (subject) representing an Application Provider (user) may access an object through the firewall.

FMT_MTD.1/JCRE SF_CARD_CONTENT_MANAGEMENT fulfils this SFR, it controls the creation of new applet instances on the card. Each time an applet instance is created, the Installer adds an entry for it in the Applet Registry

FMT_MTD.3/JCRE This SFR is fulfilled by SF_CARD_CONTENT_MANAGEMENT: it controls that only secure values are assigned as attributes of an applet instance. Invalid AIDs for the applet instances, like an AID that is already in use, are also rejected

InstG Security Functional Requirements

FDP_ITC.2/Installer This SFR is implemented by SF_CARD_CONTENT_MANAGEMENT: The SF ensures safe package loading and applet installation process. It modifies the CAP files to produce the TOE intern representation of the loaded package. It also performs coherency checks on the CAP files and verifies the export references.

FMT_SMR.1/Installer This SFR is implemented by SF_CARD_CONTENT_MANAGEMENT: The TSF is in charge of creating the applet instance that plays the role of the Applet Installation Manager.

FPT_FLS.1/Installer This SFR is fulfilled by the following SF:

- The SF_CARD_CONTENT_MANAGEMENT: is in charge of checking that all the conditions for safely installing a package or an applet instance are fulfilled during the installation procedure. If conditions cannot be verified the installation is deemed unsuccessful and either an exception is thrown or the card is frozen, depending of the failure severity. Card tearing or reset also cause an installation failure.
- SF_ATOMIC_TRANSACTIONS is in charge of rolling back to a secure state when the installation of a package or an applet instance is aborted
- This function is in charge of clearing the information contained in the packages that is not necessary for the execution of the code of the applet invoked by SF_CLEARING_OF_SENSITIVE_INFORMATION.

FPT_RCV.3/Installer This SFR is fulfilled by the following SF:

- SF_CARD_CONTENT_MANAGEMENT: In case of severe failure during package or applet installation, the card is frozen (KillCard). Such failures (for example the loading of a CAP file with an invalid format) are considered as security problems. The maintenance mode is represented by the frozen state of the card. The secure state is then reached on next card reset where Garbage Collector is launch to retrieve lost memory and where the transaction mechanism allows retrieving the initial state.

- SF_ATOMIC_TRANSACTION: The TSF is in charge of rolling back to a secure state when the installation of a package or an applet instance is aborted.

ADELG Security Functional Requirements

FDP_ACC.2/ADEL The access control policy for deletion is made by SF_CARD_CONTENT_MANAGEMENT, it controls whether the Applet Deletion Manager (subject) may delete (operation) a package or an applet instance (object).

FDP_ACF.1/ADEL The access control policy for deletion is made by SF_CARD_CONTENT_MANAGEMENT, it controls whether the Applet Deletion Manager (subject) may delete (operation) a package or an applet instance (object).

FDP_RIP.1/ADEL The TSF SF_CLEARING_OF_SENSITIVE_INFORMATION renders inaccessible the code of a deleted package and the class instances and arrays allocated by a deleted applet instance.

FMT_MSA.1/ADEL The ADEL access policy is implemented in SF_CARD_CONTENT_MANAGEMENT, this TSF keeps track of which applet instances are currently active on which logical channels. Only the Card Manager (which in [1] is identified with the Java Card RE role) is allowed to associate or remove the association between an applet instance and a logical channel. These actions are performed as part of command dispatching

FMT_MSA.3/ADEL The SF_CARD_CONTENT_MANAGEMENT enforces the assignment of restrictive values for the security attributes of the Applet Deletion policy.

FMT_SMF.1/ADEL Modifying the active applet security context is done by SF_CARD_MANAGEMENT_ENVIRONMENT, it's allowed to card manager.

FMT_SMR.1/ADEL This SFR is fulfilled by SF_CARD_MANAGEMENT_ENVIRONMENT: it keeps track of which applet instances are currently active on which logical channels. Only the Card Manager is allowed to associate or remove the association between an applet instance and a logical channel.

FPT_FLS.1/ADEL This SFR is ensured by the following TSF:

- SF_CARD_CONTENT_MANAGEMENT is in charge of checking that all the conditions for safely deleting a package or an applet instance are fulfilled before starting the deletion procedure.
- SF_ATOMIC_TRANSACTION: This TSF is in charge of rolling back to a secure state when the deletion of a package or an applet instance is aborted.
- SF_CLEARING_OF_SENSITIVE_INFORMATION: is in charge of checking that all the conditions for safely deleting a package or an applet instance are fulfilled before starting the deletion procedure.

ODELG Security Functional Requirements

FDP_RIP.1/ODEL This SFR is met by SF_CLEARING_OF_SENSITIVE_INFORMATION: This TSF renders inaccessible the code of a deleted package and the class instances and arrays allocated by a deleted applet instance.

FPT_FLS.1/ODEL The TSF SF_CLEARING_OF_SENSITIVE_INFORMATION is in charge of checking that all the conditions for safely deleting a package or an applet instance are fulfilled before starting the deletion procedure.

CarG Security Functional Requirements

Miscellaneous

FCO_NRO.2/CM During the loading phase, the SF_CARD_CONTENT_MANAGEMENT: controls card content loading, it verifies the proof of the origin of the Load File. Before to start the loading, the open checks that the user is authenticated, checks the presence of the < DAPBlock > in the Load file, requires the Security Domain Verifier to verify it.

FDP_IFC.2/CM The rule of the package loading flow control policy is specified by SF_CARD_CONTENT_MANAGEMENT: it verifies that all the loading commands are issued in the Secure Channel session. It compares the Load File Data Block Hash present in the command install for load against the received. It also requires the Dap verification of all entities committed in the loading phase, ensured by SF_DAP_VERIFICATION.

FDP_IFF.1/CM This SFR is implemented by SF_DAP_VERIFICATION, it controls the communication protocol used by the CAD and the card for transmitting packages. The SFR is also implemented in the SF_CARD_CONTENT_MANAGEMENT to ensure the access control policy for the loading of the packages.

FDP_UIT.1/CM This SFR is implemented by SF_DAP_VERIFICATION, it controls imported data from modification, deletion, insertion, replay of some of the pieces of the application sent by the CAD. The verification is made by using: Encryption and decryption operations by SF_ENCRYPTION_AND_DECRYPTION function.

FIA_UID.1/CM The Security Functionality SF_GP_DISPATCHER met this SFR: While the Card manager (ISD) or Supplementary Security domain is selected, these functions test for every command if the secure channel is open. When the secure channel is not open then only these commands are available: Get data and Initialize Update. The initialize Update returns to the user the key set version, Secure Channel identifier and the card random and the card cryptogram.

FMT_MSA.1/CM This SFR is implemented by two security functions: SF_KEY_MANAGEMENT: The TSF controls that only the CM can modify its key set and can change the card life cycle and set the default application SF_CARD_CONTENT_MANAGEMENT: This TSF controls whether the active entity has the privilege and the pre-authorization for make the Card Content Management operations, and that operation still available on the card. Its controls also that the card state allows the operations.

FMT_MSA.3/CM The TSF SF_CARD_CONTENT_MANAGEMENT provides the way to lock the Security Domain with Authorized Management privilege in order to restrict its card content management ability. This TSF provides also to disable permanently the Card Content Management operations for all entities on the card.

FMT_SMF.1/CM The TSF SF_CARD_CONTENT_MANAGEMENT controls whether the active entity has the privilege and the pre-authorization for making the Card Content Management operations - modify security attributes. -, and that operation still available on the card. Its controls also that the card state allows the operations.

FMT_SMR.1/CM The TSF SF_CARD_CONTENT_MANAGEMENT verifies that authentication is successful and the active entity has loading privilege (Authorized Management privilege) before processes any Card Content management command. The successful authentication proves the user identity and role.

FTP_ITC.1/CM Installing a new package is verified by SF_CARD_CONTENT_MANAGEMENT: the SF_GP_DISPATCHER tests if secure channel is required, and verification is made by SF_DAP_VERIFICATION.

Additional Security Functional Requirements for CM

FPT_TST.1 This SFR is supported by the following TSF:

- o SF_HARDWARE_OPERATING: At each start up, security function SF_Hardware_Operating is done. Random, DES, and CRC functional modules systematically tested: a known calculus is implemented and the result is checked. SHA, AES and ECC functional modules are tested at each start up or at first use, using the same method.
- o SF_DATA_INTEGRITY: At each start up, the entire NVM integrity, so executable code, is checked. The NVM integrity is updated after patch loading so the next starup does not rise a kill card exception.

FCO_NRO.2/CM_DAP During the loading phase, SF_DAP_VERIFICATION verifies the proof of the origin of the Load File. Before to start the loading, the open checks that the user is authenticated, checks the presence of the < DAPBlock > in the Load file, requires the Security Domain Verifier to verify it.

FIA_UAU.1/CM This SFR is implemented by the following TSF:

- o SF_GP_DISPATCHER: While the Card manager (ISD) or Supplementary Security domain is selected, these functions test for every command by SF_GP_DISPATCHER if the secure channel is open.
- o SF_ENTITY_AUTHENTICATION/SECURE_CHANNEL: When the secure channel is not open then only the command available are Get Data, Initialize Update, Select.

FIA_UAU.4/CardIssuer Present the use of Card, function implemented in SF_ENTITY_AUTHENTICATION/SECURE_CHANNEL, is given by using a RNG defined in SF_RANDOM_NUMBER.

FPT_TDC.1/CM Key set and packages when imported are consistently interpreted by implementation of SF_KEY_MANAGEMENT.

FCS_COP.1/CM-SCP The TSF SF_ENTITY_AUTHENTICATION/SECURE_CHANNEL covers this SFR. It requires the cryptographic operations for the creation and management of secure channel. The TSF

SF_ENCRYPTION_AND_DECRYPTION provides a mechanism for encrypting and decrypting the contents of a byte array.

Additional Security Functional Requirements for patch

FDP_ACC.2/Patch The SFR is implemented by the following TSF:

- SF_FRAMEWORK_APPLICATION_DISPATCHER: This TSF implements Access control policy for the Framework application,
- SF_PATCHING: The data (patch) sent to the TOE are protected in integrity thanks to a signature computed by the TOE developer with the dedicated key (JSK).

These TSF controls all access to all objects and all operations.

FDP_ACF.1/Patch The SFR is implemented by the following TSF:

- SF_PATCHING: The data (patch or locks) sent to the TOE are protected in integrity thanks to a signature computed by the TOE developer with the dedicated key (JSK).

These TSF controls all access to all objects and all operations.

FDP_UCT.1/Patch This SFR is met by the following TSF:

- SF_PATCHING: This TSF is in charge of the patch loading and user or subject must be successfully authenticated.

FDP_ITC.1/Patch The SFR is implemented by the following TSF:

- SF_PATCHING enables to load patches.
- SF_FRAMEWORK_APPLICATION_DISPACTHER enables patch loading before use phase.

FCS_COP.1/Patch Authentication cryptogram (signature computation and verification) are used by SF_PATCHING. Encrypted and decrypted data in bytes arrays are manipulated using SF_ENCRYPTION_AND_DECRYPTION. These functions call Cryptographic ones defined in previous FCS_COP operation.

FDP_UIT.1/Patch The SFR is implemented by the following TSF:

- SF_PATCHING: The data (patch or locks) sent to the TOE are protected in integrity thanks to a signature computed by the TOE developer with the dedicated key (JSK).

FAU_STG.2/Patch The SFR is implemented by the following TSF:

- SF_PATCHING: Upon request, the identification of the patch is returned.

Additional Security Functional Requirements for SmartCard Platform

FPT_RCV.4/SCP The TSF SF_DATA_COHERENCY shall ensure that reading from and writing to static and objects' fields interrupted by power loss have the property that the function either completes successfully, or for the indicated failure scenarios, recovers to a consistent and secure state.

Additional Security Functional Requirements for the applets

FCS_RNG.1

- SF_SECURITY_FUNCTIONS_OF_THE_IC: This TSF ensures that the security functionalities from the chip are provided to the software, and in particular RNG based on AIS31.
- SF_RANDOM_NUMBER: This TSF is in charge of providing random numbers.

Additional Security Functional Requirements for Runtime Verification Stack Control

FDP_ACC.2/RV_Stack This SFR is implemented by the following TSF:

- o SF_RUNTIME_VERIFIER: The SF implements a complete access control on the Stack operations.

FDP_ACF.1/RV_Stack This SFR is implemented by the following TSF:

- o SF_RUNTIME_VERIFIER: This SFR enforces the access conditions which guarantee the protection of the Stack.

FMT_MSA.1/RV_Stack This SFR is implemented by the following TSF:

- o SF_RUNTIME_VERIFIER: This TSF implements the management of the security attributes.

FMT_MSA.2/RV_Stack This SFR is implemented by the following TSF:

- o SF_RUNTIME_VERIFIER: This TSF ensures that only secure values for the attributes are accepted

FMT_MSA.3/RV_Stack This SFR is implemented by the following TSF:

- o SF_RUNTIME_VERIFIER: This TSF implements the initialisation of the attributes of the access control policy.

FMT_SMF.1/RV_Stack This SFR is implemented by the following TSF:

- o SF_RUNTIME_VERIFIER: The TSF specify the management function to modify the stack pointer. It controls the Stack and is able to change the associated parameter.

Heap Access

FDP_ACC.2/RV_Heap This SFR is implemented by the following TSF:

- o SF_RUNTIME_VERIFIER: it implements the access control to the Heap.,

FDP_ACF.1/RV_Heap This SFR is implemented by the following TSF:

- o SF_RUNTIME_VERIFIER: it ensures the access conditions to the Heap.

FMT_MSA.1/RV_Heap This SFR is implemented by the following TSF:

- o SF_RUNTIME_VERIFIER: This TSF implements the management of the modification of the security attributes of the access control to the Heap.

FMT_MSA.2/RV_Heap This SFR is implemented by the following TSF:

- o SF_RUNTIME_VERIFIER: This TSF implements the control that only security values are accepted for the security attributes..

FMT_MSA.3/RV_Heap This SFR is implemented by the following TSF:

- o SF_RUNTIME_VERIFIER: This TSF implements initialisation of the security attributes.

FMT_SMF.1/RV_Heap This SFR is implemented by the following TSF:

- o SF_RUNTIME_VERIFIER: The TSF directly controls the Heap and is able to change the associated parameter.

Transient Control

FDP_ACC.2/RV_Transient This SFR is implemented by the following TSF:

- SF_RUNTIME_VERIFIER: it implements the access control to guarantee the protection of Transient objects.

FDP_ACF.1/RV_Transient This SFR is implemented by the following TSF:

- SF_RUNTIME_VERIFIER: it implements access conditions and defines the security rules which guarantee the protection of Transient objects.

FMT_MSA.1/RV_Transient This SFR is implemented by the following TSF:

- SF_RUNTIME_VERIFIER: This TSF implements the management of the security attributes for the access control to the transient..

FMT_MSA.2/RV_Transient This SFR is implemented by the following TSF:

- SF_RUNTIME_VERIFIER: This TSF implements the condition that only secure attributes are accepted for the access control policy to the Transient.

FMT_MSA.3/RV_Transient This SFR is implemented by the following TSF:

- SF_RUNTIME_VERIFIER: This TSF controls that only restrictive values are accepted for security attributes used to enforce the transient access control policy.

FMT_SMF.1/RV_Transient This SFR is implemented by the following TSF:

- SF_RUNTIME_VERIFIER: The TSF directly controls the Transient and is able to change the associated parameter.

Additional Security Functional Requirement for Sensitive Array package

FDP_SDI.2/ARRAY The TSF SF_DATA_INTEGRITY ensures integrity of sensitive arrays. A loss of integrity increases killcard counter.

Consumer Device

Identification and authentication

FIA_UAU.1/EXT This SFR is ensured by the following TSF:

- SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FIA_USB.1/EXT This SFR is ensured by the following TSF:

- SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FIA_UAU.4/EXT This SFR is ensured by the following TSF:

- SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FIA_UID.1/MNO-SD This SFR is ensured by the following TSF:

- o SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FIA_USB.1/MNO-SD This SFR is ensured by the following TSF:

- o SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FIA_API.1 This SFR is ensured by the following TSF:

- o SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.

FIA_UID.1/EXT This SFR is ensured by the following TSF:

- o SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FIA_ATD.1/User This SFR is ensured by the following TSF:

- o SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

Communication

FDP_IFC.1/SCP This SFR is ensured by the following TSF:

- o SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.

FDP_IFF.1/SCP This SFR is ensured by the following TSF:

- o SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FTP_ITC.1/SCP This SFR is ensured by the following TSF:

- o SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FDP_ITC.2/SCP This SFR is ensured by the following TSF:

- o SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FPT_TDC.1/SCP This SFR is ensured by the following TSF:

- o SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FDP_UCT.1/SCP This SFR is ensured by the following TSF:

- o SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.
- o SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

FDP_UIT.1/SCP This SFR is ensured by the following TSF:

- SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.
- SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

FCS_CKM.1/SCP-SM This SFR is ensured by the following TSF:

- SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FCS_CKM.2/SCP-MNO This SFR is ensured by the following TSF:

- SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FCS_CKM.4/SCP-SM This SFR is ensured by the following TSF:

- SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FCS_CKM.4/SCP-MNO This SFR is ensured by the following TSF:

- SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

Security Domains

FDP_ACC.1/ISDR This SFR is ensured by the following TSF:

- SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

FDP_ACF.1/ISDR This SFR is ensured by the following TSF:

- SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

FDP_ACC.1/ECASD This SFR is ensured by the following TSF:

- SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

FDP_ACF.1/ECASD This SFR is ensured by the following TSF:

- SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

Platform Services

FDP_IFC.1/Platform_services This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.

FDP_IFF.1/Platform_services This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.

FPT_FLS.1/Platform_services This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.

Security management

FPT_EMS.1 This SFR is ensured by the following TSF:

- SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

FDP_SDI.1 This SFR is ensured by the following TSF:

- SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

FDP_RIP.1/EUICC This SFR is ensured by the following TSF:

- SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

FPT_FLS.1/EUICC This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.

FMT_MSA.1/PLATFORM_DATA This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.

FMT_MSA.1/PPR This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.

FMT_MSA.1/CERT_KEYS This SFR is ensured by the following TSF:

- SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FMT_SMF.1/EUICC This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.
- SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FMT_SMR.1/EUICC This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.

- SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

FMT_MSA.1/RAT This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.

FMT_MSA.3 This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.
- SF_SECURE_DOMAIN is in charge of managing security domains and protecting the communication by secure channel.

Mobile Network authentication

FCS_COP.1/Mobile_network This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.
- SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

FCS_CKM.2/Mobile_network This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.
- SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

FCS_CKM.4/Mobile_network This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.
- SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

SUCI

FCS_CKM.1/SUCI This SFR is ensured by the following TSF:

- SF_ACCESS is in charge of checking a correct access to the TOE and its data but also to the services of the TOE with correct operation.

FCS_CKM.4/SUCI This SFR is ensured by the following TSF:

- SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

FCS_COP.1/SUCI This SFR is ensured by the following TSF:

- SF_DATA_PROTECTION is in charge of checking or use the confidentiality and integrity of user and TSF data.

9.2.1.1 TOE Summary Specification

eUICC Security Functions

SF_ACCESS This SFR ensures the correct operation of access management for the security function.

Runtime environment Security Functions

SF_CARDHOLDER_VERIFICATION This security function is ensured by FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/APDU, FDP_RIP.1/GlobalArray, FDP_RIP.1/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL, FDP_RIP.1/TRANSIENT, FPR_UNO.1, FDP_ROL.1/FIREWALL and FDP_SDI.2/DATA security functional requirements. The TSFs behind these are implemented by API classes. The firewall security functions FDP_ACC.2/FIREWALL and FDP_ACF.1/FIREWALL shall protect the access to private and internal data of the objects.

9.2.2 Association tables of SFRs and TSS

Security Functional Requirements	TOE Summary Specification
FDP_ACC.2/FIREWALL	SF_FIREWALL, SF_CARDHOLDER_VERIFICATION
FDP_ACF.1/FIREWALL	SF_FIREWALL, SF_CARDHOLDER_VERIFICATION
FDP_IFC.1/JCVM	SF_FIREWALL
FDP_IFF.1/JCVM	SF_FIREWALL
FDP_RIP.1/OBJECTS	SF_CLEARING_OF_SENSITIVE_INFORMATION, SF_CARDHOLDER_VERIFICATION
FMT_MSA.1/JCRE	SF_FIREWALL
FMT_MSA.1/JCVM	SF_FIREWALL
FMT_MSA.2/FIREWALL_JCVM	SF_FIREWALL
FMT_MSA.3/FIREWALL	SF_FIREWALL
FMT_MSA.3/JCVM	SF_FIREWALL
FMT_SMF.1/Firewall	SF_CARD_CONTENT_MANAGEMENT
FMT_SMR.1/Firewall	SF_FIREWALL
FDP_RIP.1/ABORT	SF_CLEARING_OF_SENSITIVE_INFORMATION, SF_ATOMIC_TRANSACTION, SF_CARDHOLDER_VERIFICATION
FDP_RIP.1/APDU	SF_CLEARING_OF_SENSITIVE_INFORMATION, SF_CARDHOLDER_VERIFICATION
FDP_RIP.1/bArray	SF_CLEARING_OF_SENSITIVE_INFORMATION, SF_CARDHOLDER_VERIFICATION
FDP_RIP.1/KEYS	SF_CLEARING_OF_SENSITIVE_INFORMATION, SF_CARDHOLDER_VERIFICATION
FDP_RIP.1/TRANSIENT	SF_CLEARING_OF_SENSITIVE_INFORMATION, SF_CARDHOLDER_VERIFICATION
FDP_ROL.1/FIREWALL	SF_FIREWALL, SF_ATOMIC_TRANSACTION, SF_CARDHOLDER_VERIFICATION

Security Functional Requirements	TOE Summary Specification
FDP_RIP.1/GlobalArray	SF_CLEARING_OF_SENSITIVE_INFORMATION, SF_CARDHOLDER_VERIFICATION
FCS_CKM.1/CM-SCP	SF_KEY_GENERATION
FCS_CKM.4/CM-SCP	SF_KEY_DESTRUCTION
FCS_COP.1/Disp	SF_KEY_AGREEMENT, SF_SIGNATURE, SF_MESSAGE_DIGEST
FAU_ARP.1	SF_EXCEPTION
FDP_SDI.2/DATA	SF_DATA_PROTECTION, SF_CARDHOLDER_VERIFICATION
FPR_UNO.1	SF_UNOBSERVABILITY
FPT_FLS.1/VM	SF_ATOMIC_TRANSACTION
FPT_TDC.1/VM	SF_CARD_MANAGEMENT_ENVIRONMENT
FIA_ATD.1/AID	SF_CARD_CONTENT_MANAGEMENT
FIA_UID.2/AID	SF_FIREWALL
FIA_USB.1/AID	SF_FIREWALL
FMT_MTD.1/JCRE	SF_CARD_CONTENT_MANAGEMENT
FMT_MTD.3/JCRE	SF_CARD_CONTENT_MANAGEMENT
FDP_ITC.2/Installer	SF_CARD_CONTENT_MANAGEMENT
FMT_SMR.1/Installer	SF_CARD_CONTENT_MANAGEMENT
FPT_FLS.1/Installer	SF_CARD_CONTENT_MANAGEMENT, SF_CLEARING_OF_SENSITIVE_INFORMATION, SF_ATOMIC_TRANSACTION
FPT_RCV.3/Installer	SF_CARD_CONTENT_MANAGEMENT, SF_ATOMIC_TRANSACTION
FDP_ACC.2/ADEL	SF_CARD_CONTENT_MANAGEMENT
FDP_ACF.1/ADEL	SF_CARD_CONTENT_MANAGEMENT
FDP_RIP.1/ADEL	SF_CLEARING_OF_SENSITIVE_INFORMATION
FMT_MSA.1/ADEL	SF_CARD_CONTENT_MANAGEMENT
FMT_MSA.3/ADEL	SF_CARD_CONTENT_MANAGEMENT
FMT_SMF.1/ADEL	SF_CARD_MANAGEMENT_ENVIRONMENT
FMT_SMR.1/ADEL	SF_CARD_MANAGEMENT_ENVIRONMENT
FPT_FLS.1/ADEL	SF_CARD_CONTENT_MANAGEMENT, SF_CLEARING_OF_SENSITIVE_INFORMATION, SF_ATOMIC_TRANSACTION
FDP_RIP.1/ODEL	SF_CLEARING_OF_SENSITIVE_INFORMATION
FPT_FLS.1/ODEL	SF_CLEARING_OF_SENSITIVE_INFORMATION
FCO_NRO.2/CM	SF_CARD_CONTENT_MANAGEMENT
FDP_IFC.2/CM	SF_CARD_CONTENT_MANAGEMENT

Security Functional Requirements	TOE Summary Specification
FDP_IFF.1/CM	SF_DAP_VERIFICATION
FDP_UIT.1/CM	SF_DAP_VERIFICATION, SF_ENCRYPTION_AND_DECRYPTION
FIA_UID.1/CM	SF_GP_DISPATCHER
FMT_MSA.1/CM	SF_KEY_MANAGEMENT
FMT_MSA.3/CM	SF_CARD_CONTENT_MANAGEMENT
FMT_SMF.1/CM	SF_CARD_CONTENT_MANAGEMENT
FMT_SMR.1/CM	SF_CARD_CONTENT_MANAGEMENT
FTP_ITC.1/CM	SF_CARD_CONTENT_MANAGEMENT
FPT_TST.1	SF_CARD_CONTENT_MANAGEMENT, SF_HARDWARE_OPERATING, SF_DATA_INTEGRITY, SF_DATA_PROTECTION
FCO_NRO.2/CM_DAP	SF_DAP_VERIFICATION
FIA_UAU.1/CM	SF_DATA_INTEGRITY, SF_FIREWALL
FIA_UAU.4/CardIssuer	SF_DATA_INTEGRITY, SF_RANDOM_NUMBER
FPT_TDC.1/CM	SF_KEY_MANAGEMENT
FCS_COP.1/CM-SCP	SF_ENTITY_AUTHENTICATION/SECURE_CHANNEL, SF_ENCRYPTION_AND_DECRYPTION
FDP_ACC.2/Patch	SF_PATCHING
FDP_ACF.1/Patch	SF_PATCHING
FDP_UCT.1/Patch	SF_PATCHING
FDP_ITC.1/Patch	SF_PATCHING
FCS_COP.1/Patch	SF_PATCHING
FDP_UIT.1/Patch	SF_PATCHING
FAU_STG.2/Patch	SF_PATCHING
FPT_RCV.4/SCP	SF_DAP_VERIFICATION, SF_DATA_COHERENCY
FCS_RNG.1	SF_RANDOM_NUMBER, SF_SECURITY_FUNCTIONS_OF_THE_IC
FDP_ACC.2/RV_Stack	SF_RUNTIME_VERIFIER
FDP_ACF.1/RV_Stack	SF_RUNTIME_VERIFIER
FMT_MSA.1/RV_Stack	SF_RUNTIME_VERIFIER
FMT_MSA.2/RV_Stack	SF_RUNTIME_VERIFIER
FMT_MSA.3/RV_Stack	SF_RUNTIME_VERIFIER
FMT_SMF.1/RV_Stack	SF_RUNTIME_VERIFIER
FDP_ACC.2/RV_Heap	SF_RUNTIME_VERIFIER
FDP_ACF.1/RV_Heap	SF_RUNTIME_VERIFIER

Security Functional Requirements	TOE Summary Specification
FMT_MSA.1/RV_Heap	SF_RUNTIME_VERIFIER
FMT_MSA.2/RV_Heap	SF_RUNTIME_VERIFIER
FMT_MSA.3/RV_Heap	SF_RUNTIME_VERIFIER
FMT_SMF.1/RV_Heap	SF_RUNTIME_VERIFIER
FDP_ACC.2/RV_Transient	SF_RUNTIME_VERIFIER
FDP_ACF.1/RV_Transient	SF_RUNTIME_VERIFIER
FMT_MSA.1/RV_Transient	SF_RUNTIME_VERIFIER
FMT_MSA.2/RV_Transient	SF_RUNTIME_VERIFIER
FMT_MSA.3/RV_Transient	SF_RUNTIME_VERIFIER
FMT_SMF.1/RV_Transient	SF_RUNTIME_VERIFIER
FDP_SDI.2/ARRAY	SF_DATA_PROTECTION
FIA_UAU.1/EXT	SF_SECURE_DOMAIN
FIA_USB.1/EXT	SF_SECURE_DOMAIN
FIA_UAU.4/EXT	SF_SECURE_DOMAIN
FIA_UID.1/MNO-SD	SF_SECURE_DOMAIN
FIA_USB.1/MNO-SD	SF_SECURE_DOMAIN
FIA_API.1	SF_ACCESS
FIA_UID.1/EXT	SF_SECURE_DOMAIN
FIA_ATD.1/User	SF_SECURE_DOMAIN
FDP_IFC.1/SCP	SF_ACCESS
FDP_IFF.1/SCP	SF_SECURE_DOMAIN
FTP_ITC.1/SCP	SF_SECURE_DOMAIN
FDP_ITC.2/SCP	SF_SECURE_DOMAIN
FPT_TDC.1/SCP	SF_SECURE_DOMAIN
FDP_UCT.1/SCP	SF_DATA_PROTECTION, SF_SECURE_DOMAIN
FDP_UIT.1/SCP	SF_DATA_PROTECTION, SF_SECURE_DOMAIN
FCS_CKM.1/SCP-SM	SF_SECURE_DOMAIN
FCS_CKM.2/SCP-MNO	SF_SECURE_DOMAIN
FCS_CKM.4/SCP-SM	SF_SECURE_DOMAIN
FCS_CKM.4/SCP-MNO	SF_SECURE_DOMAIN
FDP_ACC.1/ISDR	SF_DATA_PROTECTION
FDP_ACF.1/ISDR	SF_DATA_PROTECTION
FDP_ACC.1/ECASD	SF_DATA_PROTECTION
FDP_ACF.1/ECASD	SF_DATA_PROTECTION

Security Functional Requirements	TOE Summary Specification
FDP_IFC.1/Platform_services	SF_ACCESS
FDP_IFF.1/Platform_services	SF_ACCESS
FPT_FLS.1/Platform_services	SF_ACCESS
FPT_EMS.1	SF_DATA_PROTECTION
FDP_SDI.1	SF_DATA_PROTECTION
FDP_RIP.1/EUICC	SF_DATA_PROTECTION
FPT_FLS.1/EUICC	SF_ACCESS
FMT_MSA.1/PLATFORM_DATA	SF_ACCESS
FMT_MSA.1/PPR	SF_ACCESS
FMT_MSA.1/CERT_KEYS	SF_SECURE_DOMAIN
FMT_SMF.1/EUICC	SF_ACCESS, SF_SECURE_DOMAIN
FMT_SMR.1/EUICC	SF_ACCESS, SF_SECURE_DOMAIN
FMT_MSA.1/RAT	SF_ACCESS
FMT_MSA.3	SF_ACCESS, SF_SECURE_DOMAIN
FCS_COP.1/Mobile_network	SF_ACCESS, SF_DATA_PROTECTION
FCS_CKM.2/Mobile_network	SF_ACCESS, SF_DATA_PROTECTION
FCS_CKM.4/Mobile_network	SF_ACCESS, SF_DATA_PROTECTION
FCS_CKM.1/SUCI	SF_ACCESS
FCS_CKM.4/SUCI	SF_DATA_PROTECTION
FCS_COP.1/SUCI	SF_DATA_PROTECTION

Table 16 SFRs and TSS - Coverage

TOE Summary Specification	Security Functional Requirements
SF_ACCESS	FIA_API.1, FDP_IFC.1/SCP, FDP_IFC.1/Platform_services, FDP_IFF.1/Platform_services, FPT_FLS.1/Platform_services, FPT_FLS.1/EUICC, FMT_MSA.1/PLATFORM_DATA, FMT_MSA.1/PPR, FMT_SMF.1/EUICC, FMT_SMR.1/EUICC, FMT_MSA.1/RAT, FMT_MSA.3, FCS_COP.1/Mobile_network, FCS_CKM.2/Mobile_network, FCS_CKM.4/Mobile_network, FCS_CKM.1/SUCI
SF_DATA_PROTECTION	FDP_SDI.2/ARRAY, FDP_UCT.1/SCP, FDP_UIT.1/SCP, FDP_ACC.1/ISDR, FDP_ACF.1/ISDR, FDP_ACC.1/ECASD, FDP_ACF.1/ECASD, FPT_EMS.1, FDP_SDI.1, FDP_RIP.1/EUICC, FCS_COP.1/Mobile_network, FCS_CKM.2/Mobile_network, FCS_CKM.4/Mobile_network, FCS_CKM.4/SUCI, FCS_COP.1/SUCI, FDP_SDI.2/DATA, FPT_TST.1
SF_SECURE_DOMAIN	FIA_UAU.1/EXT, FIA_USB.1/EXT, FIA_UAU.4/EXT, FIA_UID.1/MNO-SD, FIA_USB.1/MNO-SD, FIA_UID.1/EXT,

TOE Summary Specification	Security Functional Requirements
	FIA_ATD.1/User, FDP_IFF.1/SCP, FTP_ITC.1/SCP, FDP_ITC.2/SCP, FPT_TDC.1/SCP, FDP_UCT.1/SCP, FDP_UIT.1/SCP, FCS_CKM.1/SCP-SM, FCS_CKM.2/SCP-MNO, FCS_CKM.4/SCP-SM, FCS_CKM.4/SCP-MNO, FMT_MSA.1/CERT_KEYS, FMT_SMF.1/EUICC, FMT_SMR.1/EUICC, FMT_MSA.3
SF_ATOMIC_TRANSACTION	FPT_FLS.1/Installer, FPT_RCV.3/Installer, FPT_FLS.1/ADEL, FDP_RIP.1/ABORT, FDP_ROL.1/FIREWALL, FPT_FLS.1/VM
SF_UNOBSERVABILITY	FPR_UNO.1
SF_SIGNATURE	FCS_COP.1/Disp
SF_SECURITY_FUNCTIONS_OF_THE_IC	FCS_RNG.1
SF_RUNTIME_VERIFIER	FDP_ACC.2/RV_Stack, FDP_ACF.1/RV_Stack, FMT_MSA.1/RV_Stack, FMT_MSA.2/RV_Stack, FMT_MSA.3/RV_Stack, FMT_SMF.1/RV_Stack, FDP_ACC.2/RV_Heap, FDP_ACF.1/RV_Heap, FMT_MSA.1/RV_Heap, FMT_MSA.2/RV_Heap, FMT_MSA.3/RV_Heap, FMT_SMF.1/RV_Heap, FDP_ACC.2/RV_Transient, FDP_ACF.1/RV_Transient, FMT_MSA.1/RV_Transient, FMT_MSA.2/RV_Transient, FMT_MSA.3/RV_Transient, FMT_SMF.1/RV_Transient
SF_RANDOM_NUMBER	FIA_UAU.4/CardIssuer, FCS_RNG.1
SF_PATCHING	FDP_ACC.2/Patch, FDP_ACF.1/Patch, FDP_UCT.1/Patch, FDP_ITC.1/Patch, FCS_COP.1/Patch, FDP_UIT.1/Patch, FAU_STG.2/Patch
SF_MESSAGE_DIGEST	FCS_COP.1/Disp
SF_KEY_MANAGEMENT	FMT_MSA.1/CM, FPT_TDC.1/CM
SF_KEY_GENERATION	FCS_CKM.1/CM-SCP
SF_KEY_DESTRUCTION	FCS_CKM.4/CM-SCP
SF_KEY_AGREEMENT	FCS_COP.1/Disp
SF_HARDWARE_OPERATING	FPT_TST.1
SF_GP_DISPATCHER	FIA_UID.1/CM
SF_FIREWALL	FDP_ACC.2/FIREWALL, FDP_ACF.1/FIREWALL, FDP_IFC.1/JCVM, FDP_IFF.1/JCVM, FMT_MSA.1/JCRE, FMT_MSA.1/JCVM, FMT_MSA.2/FIREWALL_JCVM, FMT_MSA.3/FIREWALL, FMT_MSA.3/JCVM, FMT_SMR.1/Firewall, FDP_ROL.1/FIREWALL, FIA_UID.2/AID, FIA_USB.1/AID, FIA_UAU.1/CM
SF_EXCEPTION	FAU_ARP.1
SF_ENTITY_AUTHENTICATION/SECURE_CHANNEL	FCS_COP.1/CM-SCP

TOE Summary Specification	Security Functional Requirements
SF_ENCRYPTION_AND_DECRYPTION	FDP_UIT.1/CM, FCS_COP.1/CM-SCP
SF_DATA_INTEGRITY	FPT_TST.1, FIA_UAU.1/CM, FIA_UAU.4/CardIssuer
SF_DATA_COHERENCY	FPT_RCV.4/SCP
SF_DAP_VERIFICATION	FDP_IFF.1/CM, FDP_UIT.1/CM, FCO_NRO.2/CM_DAP, FPT_RCV.4/SCP
SF_CLEARING_OF_SENSITIVE_INFORMATION	FPT_FLS.1/Installer, FDP_RIP.1/ADEL, FPT_FLS.1/ADEL, FDP_RIP.1/ODEL, FPT_FLS.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/ABORT, FDP_RIP.1/APDU, FDP_RIP.1/bArray, FDP_RIP.1/KEYS, FDP_RIP.1/TRANSIENT, FDP_RIP.1/GlobalArray
SF_CARDHOLDER_VERIFICATION	FDP_ACC.2/FIREWALL, FDP_ACF.1/FIREWALL, FDP_RIP.1/OBJECTS, FDP_RIP.1/ABORT, FDP_RIP.1/APDU, FDP_RIP.1/bArray, FDP_RIP.1/KEYS, FDP_RIP.1/TRANSIENT, FDP_ROL.1/FIREWALL, FDP_RIP.1/GlobalArray, FDP_SDI.2/DATA
SF_CARD_MANAGEMENT_ENVIRONMENT	FMT_SMF.1/ADEL, FMT_SMR.1/ADEL, FPT_TDC.1/VM
SF_CARD_CONTENT_MANAGEMENT	FDP_ITC.2/Installer, FMT_SMR.1/Installer, FPT_FLS.1/Installer, FPT_RCV.3/Installer, FDP_ACC.2/ADEL, FDP_ACF.1/ADEL, FMT_MSA.1/ADEL, FMT_MSA.3/ADEL, FPT_FLS.1/ADEL, FMT_SMF.1/Firewall, FIA_ATD.1/AID, FMT_MTD.1/JCRE, FMT_MTD.3/JCRE, FCO_NRO.2/CM, FDP_IFC.2/CM, FMT_MSA.3/CM, FMT_SMF.1/CM, FMT_SMR.1/CM, FTP_ITC.1/CM, FPT_TST.1

Table 17 TSS and SFRs - Coverage

10 Rationale table with objectives defined in ANSSI-CC-Note 06 [NOTE6]

TOE Security Objectives	Note 06 Security Objectives	Comment
O.SID	O.TOE_Identification	TOE is identified with its package and patch
O.LOAD	O.Secure_Load_ACode	Ensures secure Package loading
O.INSTALL	O.Secure_AC_Activation	Ensures secure activation or none installation in case of exception.
O.PATCH_LOADING	O.Secure_Load_ACode and O.TOE_Identification	O.PATCH_LOADING ensures trustable identification and authentication (static signature) data of the loaded patch. The data to be loaded are encrypted and the patch integrity is checked.

Table 18: Security Objectives Vs Note 06 Objectives