SECURITY TARGET LITE
DRAGONFLY V4.0

ISSUE: 1

Issue Date	Author	Status	Purpose
February 2016	Sarra MESTIRI	Version 1	• **ST401 4748 Issue 3 sanitising for publication according reference 2006-04-004**

Table of contents

List of tables

List of figures

1 PREFACE

1.1 OBJECTIVES OF THE DOCUMENT

The objective of this document is to present the security target Lite of the dragonFly v4.0 an Open Platform with the DESFire. The dragonFly v4.0 enables the deployment of new security services on this open platform. Applications will be deployed on this certified open platform either by the Mobile Network Operators or by third parties. These applications can be loaded before issuance or in post-issuance when the product is already in the field.

The platform includes also necessary developments for DESFire application.

The basis for this composite evaluation is the evaluation of FlyBuy open Platform with DESFire and the hardware plus the cryptographic library.

This Security Target aims to satisfy the requirements of Common Criteria level EAL4 augmented with AVA_VAN.5, ALC_FLR.3 and ALC_DVS.2 in defining the security enforcing functions of the Target Of Evaluation and describing the environment in which it operates.

1.2 SCOPE OF THE DOCUMENT

This document describes the Security Target Lite for the dragonFly v4.0, Open Platform with DESFire, which is running on a Virtual Machine.

This Security Target covers the development of the dragonFly v4.0 which is able to receive and manage different types of applications:

- Basic: applications that do not require certificate, with no assets to protect. This is the case for fidelity applications, Information-on-demand (IOD) applications, etc.
- Secure: applications that require a Common criteria certificate.

This card is consistent with the Java Card 3.0.4 Classic Edition specifications, as well as the GlobalPlatform 2.2 specification.

The objectives of this Security Target are:

- To describe the Target of Evaluation (TOE), its life cycle and to position it in the smart card life cycle.
- To describe the security environment of the TOE including the assets to be protected and the threats to be countered by the TOE and by the operational environment during the platform active phases.
- To describe the security objectives of the TOE and its supporting environment in terms of integrity and confidentiality of sensitive information. It includes protection of the TOE (and its documentation) during the platform active phases.
- To specify the security requirements which include the TOE functional requirements, the TOE assurance requirements and the security requirements for the environment.

- To describe the summary of the TOE specification including a description of the security functions and assurance measures that meet the TOE security requirements.
- To present evidence that this ST is a complete and cohesive set of requirements that the TOE provides on an effective set of IT security countermeasures within the security environment, and that the TOE summary specification addresses the requirements.

1.3 RELATED DOCUMENTS

[1] "Common Criteria for information Technology Security Evaluation, Part 1: Introduction and general model", September 2012, Version 3.1 revision 4.

[2] "Common Criteria for information Technology Security Evaluation, Part 2: Security Functional requirements", September 2012, Version 3.1 revision 4.

[3] "Common Criteria for information Technology Security Evaluation, Part 3: Security Assurance requirements", September 2012, Version 3.1 revision 4.

[4] “Composite product evaluation for Smart Cards and similar devices”, April 2012, Version 1.2, CCDB-2012-04-001.

[5] PP SUN Java Card™ System Protection Profile Open Configuration V3.0, May 2012.

[6] "Java Card - API" Application Programming Interfaces, Classic Edition, Version 3.04, September 2011, Sun Microsystems.

[7] "Java Card – JCRE” Runtime Environment Specification, Classic Edition, Version 3.04, February 23, 2009, Sun Microsystems.

[8] "Java Card - Virtual Machine Specifications" Classic Edition, Version 3.04, September 06, 2011, Sun Microsystems.

[9] GlobalPlatform Card Specification – Version 2.2.1 – January 2011.

[10] GlobalPlatform Card Mapping Guidelines of existing GP v2.1.1 implementations on v2.2.1 – Version 1.0.1 – January 2011.

[11] GlobalPlatform Card Confidential Card Content Management – Card Specification v 2.2 – Amendment A – Version 1.0.1 – January 2011.

[12] GlobalPlatform Card UICC Configuration – Version 1.0.1 – January 2011.

[13] GlobalPlatform Card Contactless Services Card Specification v 2.2 – Amendment C Version 1.1– April 2013.

[14] Visa GlobalPlatform 2.1.1 Card Implementation Requirements – Version 2.0 – July 2007.

[15] "Identification cards - Integrated Circuit(s) Cards with contacts, Part 6: Inter industry data elements for interchange", ISO / IEC 7816-6 (2004).

[16] FIPS PUB 46-3 “Data Encryption Standard”, October 25, 1999, National Institute of Standards and Technology

[17] FIPS PUB 81 “DES Modes of Operation”, December, 1980, National Institute of Standards and Technology

[18] FIPS PUB 140-2 “Security requirements for cryptographic modules”, May 2001, National Institute of Standards and Technology

[19] FIPS PUB 180-3 “Secure Hash Standard”, October 2008 , National Institute of Standards and Technology

[20] FIPS PUB 186-3 “Digital Signature Standard (DSS)”, June 2009, National Institute of Standards and Technology

[21] FIPS PUB 197, “The Advanced Encryption Standard (AES)”, November 26, 2001, National Institute of Standards and Technology

[22] SP800_90 “Recommendation for Random Number Generation Using Deterministic Random Bit Generators (Revised)”, March 2007, National Institute of Standards and Technology

[23] ANSI X9.31 “Digital Signatures using Reversible Public Key Cryptography for the Financial Services Industry (rDSA)”, 1998, American National Standards Institute

[24] ISO/IEC 9796-1, Public Key Cryptography using RSA for the financial services industry", annex A, section A.4 and A.5, and annex C (1995)

[25] ISO/IEC 9797-1, “Information technology — Security techniques — Message Authentication Codes (MACs) — Part 1: Mechanisms using a block cipher”, 1999, International Organization for Standardization

[26] PKCS#1 The public Key Cryptography standards, RSA Data Security Inc. 2003

[27] IEEE Std 1363a-2004, “Standard Specification of Public Key Cryptography – Amendment 1: Additional techniques”, 2004, IEEE Computer Society

[28] IC Platform Protection Profile, Version 1.0, reference BSI-PP-0035 (15.06.2007).

[29] SLE 97/ SLC 14 Family production and Personalization User’s Manual May 10, 2012

[30] (U)SIM Java Card Platform Protection Profile Basic Configuration. ANSSI-CC-PP 2010/04.

[31] 3GPP TS 21.111 (v11.0.1, Rel-11): USIM and IC card requirements

[32] 3GPP TS 22.038 (v8.0.1, Rel-8): USIM Application Toolkit (USAT) - Stage 1

[33] 3GPP TS 23.040 (v8.6.0, Rel-8): Technical realization of the Short Message Service (SMS)

[34] 3GPP TS 23.041 (v7.0.0, Rel-7): Technical realization of Cell Broadcast Service (CBS)

[35] 3GPP TS 23.048 (v5.9.0, Rel-5): Security Mechanisms for the (U)SIM application toolkit; Stage 2

[36] 3GPP TS 31.048 (v5.0.1, Rel-5): Test of (U)SAT security

[37] 3GPP TS 31.101 (v9.1.2, Rel-9): UICC-Terminal interface; Physical and Logical Characteristics

[38] 3GPP TS 31.102 (v8.17.0, Rel-8): Characteristics of the USIM Application

[39] 3GPP TS 31.103 (v10.101, Rel-10): Characteristics of the ISIM Application

[40] 3GPP TS 31.111 (v9.8.0, Rel-9): USIM Application Toolkit (USAT)

[41] 3GPP TS 31.115 (v11.0.0, Rel-11): Secured packet structure for (U)SIM Toolkit applications

[42] 3GPP TS 31.116 (v11.0.0, Rel-11): Remote APDU Structure for (U)SIM Toolkit applications

[43] 3GPP TS 31.122 (v9.2.0, Rel-9): USIM conformance test (card side)

[44] 3GPP TS 31.130 (v11.0.0, Rel-8): (U)SIM Application Programming Interface; (U)SIM API for Java™ Card

[45] 3GPP TR 31.900 (v11.0.0, Rel-11): SIM/USIM Internal and External Inter-working Aspects

[46] 3GPP TS 31.919 (v8.0.0, Rel-8): 2G/3G Java Card™ API based applet interworking

[47] 3GPP TS 33.102 (v8.6.0, Rel-8): 3G Security; Security architecture

[48] 3GPP TS 33.105 (v11.0.0, Rel-11): Cryptographic algorithm requirements

[49] 3GPP TS 35.205 (v11.0.0, Rel-11): Specification of the MILENAGE Algorithm Set

[50] 3GPP TS 42.017 (v4.0.0, Rel-4): SIM functional characteristics

[51] 3GPP TS 42.019 (v5.1.0, Rel-5): SIM API for Java Card™ - Stage 1 -

[52] 3GPP TS 43.019 (v6.0.0, Rel-5): Subscriber Identity Module Application Programming Interface; (SIM API) for Java Card™; Stage 2

[53] 3GPP TS 51.011 (v4.15.0, Rel-4): Specification of the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface

[54] 3GPP TS 51.014 (v4.5.0, Rel-4): Specification of the SIM Application Toolkit for the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface

[55] 3GPP TS 51.017 (v4.2.0, Rel-4): Test of SIM-ME interface (card side)

[56] ETSI TS 101 220 (v8.4.0, Rel-8): Application Identifiers for telecommunications

[57] ETSI TS 102 124 (v6.1.0, Rel-6): Transport Protocol for CAT Applications - Stage 1

[58] ETSI TS 102 127 (v6.13.0, Rel-6): Transport Protocol for CAT applications; Stage 2

[59] ETSI TS 102 151 (v6.0.0, Rel-6): Measurement of Electromagnetic Emission of SIM cards

[60] ETSI TS 102 221 (v9.2.0, Rel-9): UICC-Terminal interface; Physical and logical characteristics

[61] ETSI TS 102 222 (v7.1.0, Rel-7): Administrative Commands for telecommunications applications

[62] ETSI TS 102 223 (v9.4.0, Rel-9): Card Application Toolkit

[63] ETSI TS 102 224 (v8.0.0, Rel-8): CAT security – Stage 1

[64] ETSI TS 102 225 (v11.0.0, Rel-11): Secured packet structure for UICC applications

[65] ETSI TS 102 226 (v11.0.0, Rel-11): Remote APDU Structure for UICC based Applications

[66] ETSI TS 102 240 (v9.1.0, Rel-9): UICC Java Card™ API - Stage 1

[67] ETSI TS 102 241 (v9.2.0, Rel-9): UICC Java Card™ API - Stage 2

[68] ETSI TS 102 613 (v9.2.0, Rel-9): UICC – Contactless Front-end (CLF) Interface – Part 1: Physical and data link layer characteristics

[69] ETSI TS 102 622 (v9.4.0, Rel-9): UICC – Contactless Front-end (CLF) Interface – Host Controller Interface (HCI)

[70] ETSI TS 102 705 (v9.2.0, Rel-9): UICC Application Programming Interface for Java Card™ for Contactless Applications

[71] ETSI TS 131.111, Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Universal Subscriber Identity Module (USIM) Application Toolkit (USAT) (Release 6)

[72] ETSI TS 131.130, Digital cellular telecommunications system (Phase 2+);Universal Mobile Telecommunications System (UMTS); (U)SIM Application Programming Interface (API);(U)SIM API for Java Card (3GPP TS 31.130 version 6.6.0 Release 6)

[73] CERTIFICATION OF APPLICATIONS ON “OPEN AND ISOLATING PLATFORM, Paris, the 4th March 2011. Reference : ANSSI-CCNOTE/10EN.01deW3

[74] Security Target Lite M5072 including optional Software Libraries RSA - EC – Toolbox, Common Criteria CCv3.1 EAL5 augmented (EAL5+) Version 0.1 Date 2014-09-01

[75] Security Target Lite. Mifare DESFIRE EV1, MF31CD81 Rev1.5-10 May 2011.NXP Semiconductors.

[76] Mifare DESFIRE EV 1 Compliance and Robustness Rules v2.1

[77] GlobalPlatform Card technology Card Specifiction v2.2 – Amend E

1.4 ABBREVIATIONS AND NOTATIONS

1.4.1 Abbreviations

AES	Advanced Encryption Standard
AID	Applet Identifier
APDU	Application Protocol Data Unit
API	Application Programmer Interface
APSD	Application Provider Security Domain
BIOS	Basic Input/Output System
CASD	Controlling Authority Security Domain
CC	Common Criteria
CM	Card Manager
CPLC	Card Production Life Cycle
DAP	Data Authentication Pattern
DES	Cryptographic module "Data Encryption Standard"
EAL	Evaluation Assurance Level
EC	Elliptic Curves
EEPROM	Electrically Erasable and Programmable Read Only Memory
ES	Embedded Software
FAT	File Allocation Table
GP	Global Platform
IC	Integrated Circuit
ISD	Issuer Security Domain
IT	Information Technology
JCP	Java Card Platform
JCRE	Java Card Runtime Environment
OSP	Organizational Security Policy
PP	Protection Profile
RNG	Random Number Generation

ROM	Read Only Memory
RSA	Cryptographic module "Rivest, Shamir, Adleman"
SF	Security Function
SFP	Security Function Policy
SHA-1	Cryptographic module "Secure hash standard"
ST	Security Target
TOE	Target of Evaluation.
TSC	TSF Scope of Control
TSF	TOE Security Functions
TSP	TOE Security Policy
VASD	Validation Authority Security Domain
VM	Virtual Machine

1.4.2 Notations

Applet — Application which can be loaded and executed with the environment of the Java Card platform

Card Issuer — Entity that owns the card and is ultimately responsible for the behavior of the card

Card Manager — Main entity which represents the issuer and supervises the whole services available on the card. The Card Manager entity encompasses the Open and the Issuer Security domain.

DAP — Part of the Load File used for ensuring authenticity of the Load File. The DAP is the signature of the Load File Data Block Hash and is provided during the loading.

Delegated Management
Pre-authorized Card Content Management performed by an approved Application Provider.

Issuer Security Domain
The primary on-card entity providing support for the control, security, and communication requirements of the Card Issuer.

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.

OPEN — Part of the Card Manager entity which has the responsibilities to provide an API to applications, command dispatch, Application selection, logical channel management, and Card Content management. The OPEN also manages the installation of applications loaded to the card. The OPEN is responsible for enforcing the security policy defined for Card Content management.

Receipt — A cryptographic value provided by the card as proof that a Delegated Management operation has been successfully done. The receipt is generated on card by the Issuer Security Domain.

Security Domain	On-card entity providing support for the control, security, and communication requirements of an off-card entity (e.g. the Card Issuer, an Application Provider or a Controlling Authority).
Token	A cryptographic value provided by a Card Issuer as proof that a Delegated Management operation has been authorized. This signature is generated by the Card Issuer and used to enforce the Card Issuer control over the Card Content Management operations. The Token is verified on card by Issuer Security Domain.

2 Security Target Introduction

This Security Target (ST) is the foundation for the Evolutionary (U)SIM Java Card ™ platforms Composition Scheme. As a unique combination of Common Criteria scheme and mobile operator Validation Scheme, it creates the necessary conditions of trust for delivering new types of applications for the mobile environment. Security demanding applications such as payment applications, mobile-TV applications, identity applications (hereafter referenced as Secure Applications) are now loaded onto EAL4+ certified smart cards embedding (U)SIM Java Card Platforms along with validated applications requiring less security (hereafter Basic applications) and this during the post-issuance of the Java Card ™ platforms.

This advance opens the door for new usage models of Mobile Network Operators (MNOs) (U)SIM cards, compatible with increasing security demand. With help of the Common Criteria standard, it delivers for the first time in the mobile ecosystem, tangible benefits for all value-chain actors, including end-users.

In the following, the (U)SIM Java Card platform will be called (U)SIM platform for short and the smart card product embedding the (U)SIM platform will be called (U)SIM card.

2.1 At the Edge of New Mobile Services

This Security Target takes place in the context of the explosion of services provided by MNOs which intends to open (U)SIM card to new applications.

In order to ensure end users and application providers trust, mandatory to guarantee the success of these new services, the evolutionary certification scheme is built in accordance with the industrial and market constraints. This work starts by defining the security rules required for such an open (U)SIM Java Card platform. This is the goal of the present Security Target.

This (U)SIM platform provides services to the applications to be embedded in the platform (including correct execution of the applets), and also services for secure management of the applications.

Security Services to Applications

The TOE offers to applications a panel of security services in order to protect application data and assets:

- Confidentiality and integrity of cryptographic keys and associated operations. Cryptographic operations are protected, including protection against observation or perturbation attacks. Confidentiality and integrity of cryptographic keys and application data are guaranteed at all time during execution of cryptographic operations.
- Confidentiality and integrity of authentication data. Authentication data are protected, including protection against observation or perturbation attacks. Confidentiality and integrity of authentication data and application data are guaranteed at all time during execution of authentication operations.
- Confidentiality and integrity of application data among applications. Applications belonging to different contexts are isolated from each other. Application data are not accessible by a normal or abnormal execution of another Basic or Secure application.

- Application code execution integrity. The Java Card VM and the “applications isolation” property guarantee that the application code is operating as specified in absence of perturbations. In case of perturbation, this TOE security feature must also be valid.

DESFire

In addition to services provided by the IC and the platform, the TOE implements the DESFire module.

The DESFire module is designed to be accessed through two interfaces:

- Host Interface: It is the interface to the Card OS to manage the life cycle and the content of a MIFARE Virtual Card, to transfer MIFARE messaging such as MIFARE DESFire commands, to manage the state and to retrieve the capabilities of the DESFire module.
- Contactless Interface: It is the interface to transfer commands exchanged between a MIFARE contactless terminal and a Virtual Card.

Application Management

The TOE offers additional security services for applications management, relying on the GlobalPlatform framework:

- The MNO as Card issuer is initially the only entity authorized to manage applications (loading, instantiation, deletion) through a secure communication channel with the card, based on SMS or BIP technology. However, the MNO can grant these privileges to the AP through the Delegated Management functionality of GP.
- Before loading, all applications are verified by a validation laboratory for the Basic applications, or by an ITSEF for the secure applications. All loaded applications are associated at load time to a Verification Authority (VA) signature (Mandated DAP) that is verified on card by the on-card representative of the VA prior to the completion of the application loading operation and prior to the instantiation of any applet defined in the loaded application.
- Application Providers personalize their applications and Security Domains (APSD) in a confidential manner. Application Providers have Security Domain key sets enabling them to be authenticated to the corresponding Security Domain and to establish a trusted channel between the TOE and an external trusted device. These Security Domains key sets are not known by the Card issuer.

Note that Application Management of Secure or Basic applets can be carried out onto the TOE either by direct access to the TOE or over-the-air (OTA) through the mobile network, without physical manipulation of the TOE and in a connected environment.

The TOE offers two physical interfaces: ISO7816 and SWP interfaces.

2.2 ST REFERENCE

Title: Security Target DRAGONFLY V4.0

Name: dragonFly v4.0

Oberthur Technologies registration: FQR 401 4748

Version: Issue 3

Date of issue: February 2016

Authors: Oberthur Technologies

2.3 TOE REFERENCE

TOE Name	**dragonFly v4.0**
Internal reference	**USIM V4.0 NFC FlyBuy 4.0 on SLE97CNFX1M50PE**
Code / Hardware Identification	**412691**
Label PVCS CODE	**USIM_V31_DF4_SLE97_REVB_V03**
IC reference	**SLE97CNFX1M50PE**
IC Security Target Lite	**Security Target Lite M5072 including optional Software Libraries RSA - EC – Toolbox, Common Criteria CCv3.1 EAL5 augmented (EAL5+) Version 0.1** **Date 2014-09-01**
IC Certificate	**BSI-DSZ-CC-0946-2014**

Table 1: TOE References

The link between the IC reference as expressed in the IC ST lite -M5072- and the SLE97CNFX1M50PE is expressed in the M5072_SLE97_ProductList_2014-07-10. Table 1.2 Chapter 1.5.

The IC used includes Mifare Classic optional module and the RF library developed by Infineon. It doesn't include any other optional Toolbox.

2.3.1 TOE Identification

In order to assure the authenticity of the card, the product identification shall be verified by analysing:

- The ATR:

  3B 9F 96 80 3F C7 00 80 31 E0 73 FE 21 1F 64 **41 26 91** 00 82 90 00

  Where **41 26 91** is the SAAAAR code.

- The response of the command GET DATA:

Command : 80 CA 9F 7F 2D
Output Data : 9F 7F 2A **48 30 52 03 82 31 53 17 32 78** 00 00 00
: 00 00 00 00 00 00 00 00 00 00 00 00 00 xx xx xx
: xx xx xx xx xx xx xx xx xx 00 00 00 00

Status : 90 00

The meaning of the following bytes in the response of the command is:

- ⇨ FAB_ID : **48 30**
- ⇨ IC_ID : **52 03**
- ⇨ OS_ID : **82 31**
- ⇨ OS_Release_Date : **53 17**
- ⇨ OS_Release_Level : **32 78**

For the Card Manager, the following instruction can be used to retrieve the Card Manager identifier and the proprietary installation parameters

After selection of the ISD, the following GET DATA can be sent (no need for opening a Secure Channel) :

Command : 80 CA DF 6C
Status : 90 00
Output Data have to be interpreted following this table :

Field	Content	Remark		
Tag value	DF 6C			
Length	14			
Card Manager version	47 4F 50 20 52 65 66 20 56 32 31 2E 30 34 2E 30 32	In ASCII: “GOP Ref V21.06.01”		
Other data	2F	In ASCII: “/”		
1st byte value of TAG ‘7F’ of CM Install parameter	XX	b8	X	Lock to Enable length of hash to SHA-256 in LOAD file
		b7	X	RFU
		b6	X	Lock to disable the API Encrypt (SecureChannel API) (value 1 = API Encrypt is disabled)
		b5	X	Lock to enable Memory Resource Management
		b4	X	CASD RSA Onboard Key Generation
		b3	X	Lock to Disable DELETE SD HIERARCHY (Amd C)

		b2	X	RFU
		b1	0	Lock to enable the support of SSD with Authorized Management. In this configuration, AM is disabled (see 2.3.2.3)
2nd byte value of TAG '7F' of CM Install parameter	YY	b8	X	Lock to set tag 'B6' as optional in Token Computation and Verification
		b7	X	Lock to enable the behavior of SET STATUS as described in UICC Config V2 (instead of V1)
		b6	X	Lock to enable CASD extended RSA key length
		b5	X	Lock to enable Simple DES
		b4	X	RFU
		b3	X	RFU
		b2	X	Lock to Enable ISIS specification
		b1	1	Lock to disable Amd B features. In this configuration AMD is disabled (see 2.3.2.4)

2.3.2 Configuration Identification

2.3.2.1 Mandated DAP

The configuration evaluated in the present ST, is the one with mandated DAP.

To identify the configuration with or without MANDATED DAP the GET STATUS command (see [12]) should be used to retrieve information on installed Security Domain(s):

CLA	INS	P1	P2	Lc	Data	Le
80	F2	40	00 or 01	02	4F 00	Xx

Where P2 means:

- '00': Get first or all occurrence(s)
- '01': Get next occurrence(s)

Note: in order to process the GET STATUS command, the ISD must selected and a Secure Channel (SCP02) must be opened first (see [12] for details).

Response data field is formatted as follow:

Name	Length	Value
Length of Application AID	1	'05'-'10'
Application AID	5-16	'xxxxx…'
Life Cycle State	1	'xx' (see Table 2 and Table 3)
Privileges (byte 1)	1	'xx' (see Table 4)

b8	b7	B6	b5	b4	b3	B2	b1	Meaning
0	0	0	0	0	0	1	1	INSTALLED
0	0	0	0	0	1	1	1	SELECTABLE
0	0	0	0	1	1	1	1	PERSONALIZED
1	0	0	0	-	-	1	1	LOCKED

Table 2: Security Domain Life Cycle Coding

b8	b7	B6	b5	b4	b3	b2	b1	Meaning
0	0	0	0	0	0	1	1	OP_READY
0	0	0	0	0	1	1	1	INITIALIZED
0	0	0	0	1	1	1	1	SECURED
0	1	1	1	1	1	1	1	CARD_LOCKED
1	1	1	1	1	1	1	1	TERMINATED

Table 3: Card Life Cycle Coding

b8	b7	b6	B5	b4	b3	B2	b1	Meaning
1	-	-	-	-	-	-	-	Security Domain
1	1	-	-	-	-	-	0	DAP Verification
1	-	1	-	-	-	-	-	Delegated Management
-	-	-	1	-	-	-	-	Card Lock
-	-	-	-	1	-	-	-	Card Terminate
-	-	-	-	-	1	-	-	Card Reset
-	-	-	-	-	-	1	-	CVM Management
1	1	-	-	-	-	-	1	**Mandated DAP Verification**

Table 4: Privileges (byte 1)

A successful execution of the command shall be indicated by status bytes '90' '00'.

The command may return the following warning condition: '63' '10' More data available. If so, a subsequent GET STATUS [get next occurrence(s)] may be issued to retrieve additional data.

If the AID of the Security Domain with Mandated DAP Privilege is known, the command to perform for checking that this SD is present on the card is the following:

Command : 80 F2 40 00 [Length of Input] 4F [Length of SD AID] [AID of SD]

Output Data : [Length of SD AID] [AID of SD] [Life Cycle State]$_{1\ byte}$ C1
Status : 90 00

For instance, if AID of SD MD = A0 00 00 00 01 23 45 67, the command will be:

Command : 80 F2 40 00 0A 4F 08 A0 00 00 00 01 23 45 67
Output Data : 08 A0 00 00 00 01 23 45 67 0F **C1**
Status : 90 00

C1 = Privilege of SD with Mandated DAP verification

2.3.2.2 Card Lock

The card must be in a locked configuration.

This can be verified with the following command which shall send 0xFF as output data.

Command	: A0 BC 00 00 01
Output Data	: FF
Status	: 90 00

Note: This command must be sent to the GSM application (default selected after POWER ON). The ADM1 PIN must be verified before using the described command.

The TOE guidance references are included in the chapter 2.5.12.1.

2.3.2.3 Authorized Management Lock

The Authorized Management is not supported in the claimed PP, so it is deactivated on the TOE.

In order to check that the card does not allow SSD with Authorized Management privilege, the command GET DATA “Card Manager Release” should be used.

The byte ‘XX’ in the response data (see §2.3.1) is the 1st byte value of the TAG ‘7F’ of the Card Manager Install parameter: bit 1 shall be set to ‘0’.

2.3.2.4 Amendment B Lock

The Amendment B is not supported in this ST so it is deactivated on the present TOE.

In order to check that the card does not allow amendment B features, the command GET DATA “Card Manager Release” should be used.

The byte ‘YY’ in the response data (see §2.3.1) is the 2nd byte value of the TAG ‘7F’ of the Card Manager Install parameter: bit 1 shall be set to ‘1’.

2.4 TOE Overview

This section presents the architecture and common usages of the Target of Evaluation (TOE).

This Security Target focuses on the security requirements for the (U)SIM Java Card platform including the Smart Card Platform (SCP, the combination of the IC and the OS) and the DESFire, the TOE content details are expressed in the chapter 2.5.

The Figure 1 expresses the architecture in high level view. The TOE Security Function (TSF) is restricted to the parts included in the red dash lines.

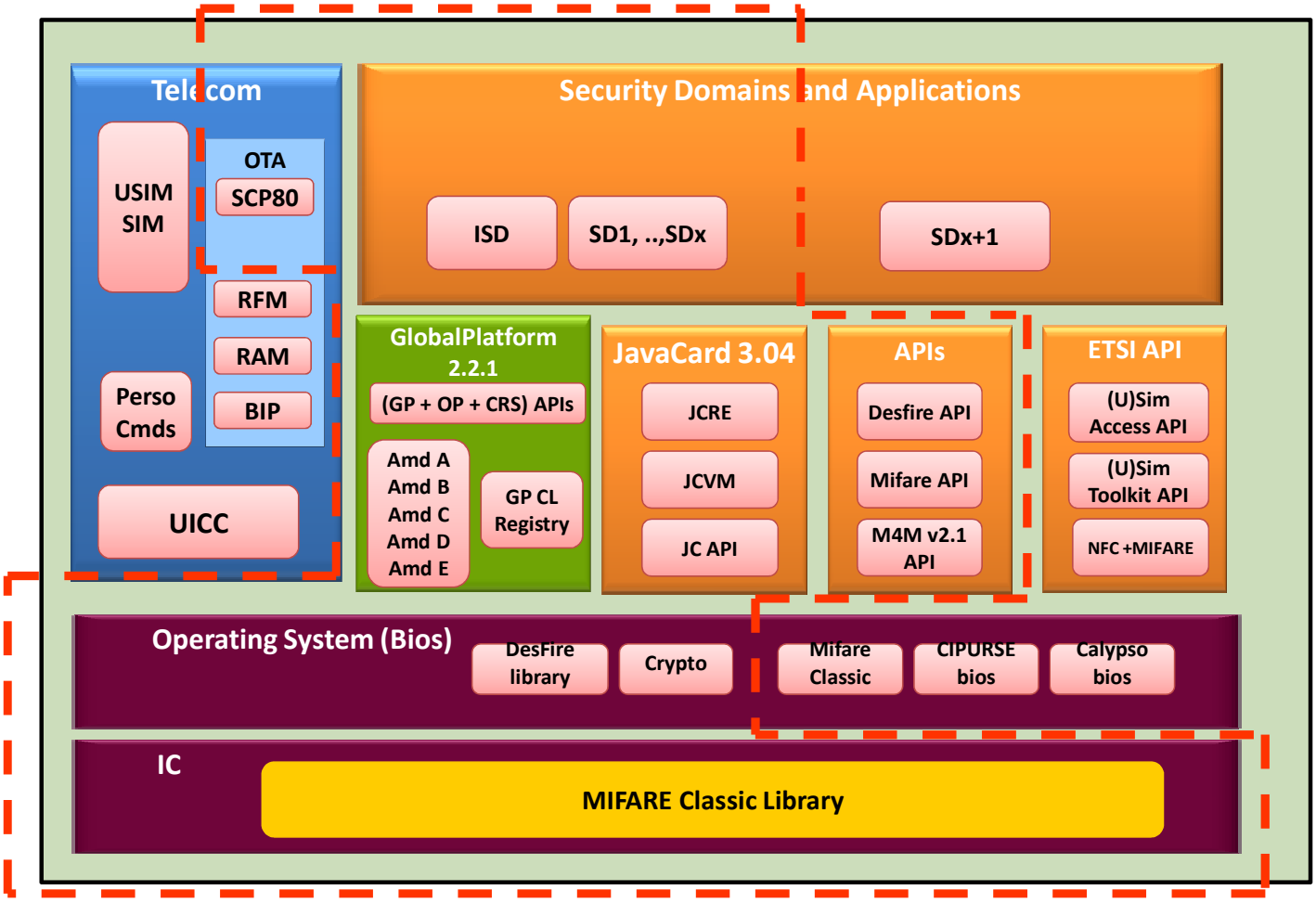


Figure **1: dragonFly V4.0 Architecture**

The (U)SIM card is intended to be plugged in a mobile phone or other mobile devices to provide services to an end user.

The Target of Evaluation (TOE) is composed of the following bricks:

- A Java Card System according to [8] which manages and executes applications called applets. It also provides APIs [6] to develop applets on top of it, in accordance with Java Card ™ specifications (applets are not included in the present security evaluation).
- GlobalPlatform (GP) packages, which provides a common and widely used interface to communicate with a smart card and manage applications in a secure way, in accordance with [9] specifications,
- ETSI APIs, which provides ways to specifically interact with (U)SIM applications, according to [44] specifications.
- An Operating system as an interface between hardware and software like VM and APIs and cryptographic services.

- Usim functionalities, which provide all the functionalities described in [31][32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55] [56] [59] [60] [61] [62] [63] [64] [65] [66] [67] [68] [69] [70] [71] [72]: UICC commands, Network authentication, OTA commands, etc…
- Perso Cmds, Personalisation Commands, which provide a set of proprietary commands for personalization.
- BIP protocol, BIP Over CAT-TP protocol describes in the specifications [57] [58].
- DESFire specific Functions to allow integrity and confidentiality of the DESFire application Data and Code.

The product can host additional applets loaded in pre-issuance phase. The additional applications are not included in the TOE.

2.5 TOE Description

The global architecture of the **DragonFLY V4.0** is presented in Figure 1: DragonFLY V4.0. Architecture. The TOE contains all the grey part, and the TSF is limited in the red lines. Some basic applications are already included in the TOE. At post issuance, the TOE can have additional applications not represented in the figure 1.

Each TOE block is detailed in the following paragraphs.

2.5.1 (U)SIM Functionality

The (U)SIM specifications considered in this document are the Release 7 versions of the ETSI 3GPP specifications.

Data exchange between the TOE and the mobile network, including applications downloading are enforced through a communication channel based on SMS or CAT TP over BIP technology.

The (U)SIM supports at least the following APIs:

- The UICC API [[66][67]] is an extension of the SIM API [32] which provides the means for the applications to access the smart card file system, to subscribe in order to receive the events of the common application toolkit framework, to handle information received and to send proactive commands.
- The (U)SIM API [44] extends the UICC API to provide features related to the 3G: it provides the means for applets to get access to the files of the (U)SIM, to register to the events defined in the USAT specification, etc.

All (U)SIM functionalities are part of the TOE.

2.5.2 Bearer Independent Protocol (BIP)

The BIP technology is an Over-The-Air (OTA) technology to exchange data between a (U)SIM card on a mobile phone and remote servers. It will replace the SMS technology as a data bearer for mobile phones.

The BIP technology relies on high speed communication protocols such as GPRS, EDGE, UMTS, Bluetooth, USB 2.0 and infrared of new generation mobile phones (2.5G and 3G). Therefore, the communication is faster and more reliable than the SMS channel. Local communication channels with the SIM (Bluetooth, USB, infrared) are not supported in this Security target.

This technology is a part of the TOE. It is specified in 3GPP specifications. In particular, [34], [35] and [36] are implemented.

The BIP technology does not offer any security function.

2.5.3 Java Card Platform

The Java technology, embedded on the TOE, combines a subset of the Java programming language with a runtime environment optimized for smart cards and similar small-memory embedded devices [8].

The Java Card ™ platform is a smart card platform enabled with Java Card ™ technology (also called, for short, a "Java Card"). This technology allows for multiple applications to run on a single card and provides facilities for secure interoperability of applications. Applications running on the Java Card platform ("Java Card applications") are called applets.

The TOE is compliant with the version of the Java Card platform Classic edition 3.0.4 specified in [8], [7] and [6]. It includes the Java Card Virtual Machine (Java Card VM), the Java Card Runtime Environment (Java Card RE) and the Java Card Application Programming Interface (Java Card API), detailed in the next paragraph. As the terminology is sometimes confusing, the term "Java Card System" has been introduced in [5] to designate the set made of the Java Card RE, the Java Card VM and the Java Card API. The Java Card System provides an intermediate layer between the operating system of the card and the applications. This layer allows applications written for one smart card platform enabled with Java Card technology to run on any other such platform.

The Java Card VM is a bytecode interpreter embedded in the smart card. The Java Card RE is responsible for card resource management, communication, applet execution, on-card system and applet security.

The TOE is configured so that an applet can be downloaded and installed on it, even after the smart card has been issued to the Cardholder. This allows MNOs as Card issuers to dynamically respond to customers' needs. For instance, if the Card issuer decides to upgrade some of the applications offered to the customer, this could be made without issuing a new card. Moreover, applications from different vendors can coexist in a single card, and they can even share information between each other. Since a smart card application is usually intended to store highly sensitive information, the sharing of that information must be carefully controlled.

Applet isolation is achieved through the Java Card Firewall mechanism defined in [7]. This mechanism confines an applet to its own designated memory area. Thus, each applet is prevented from accessing fields and operations related to objects owned by other applets, unless those applets provide a specific interface (shareable interface) for that purpose. This access control policy is enforced at runtime by the Java Card VM.

However, applet isolation cannot be entirely granted by the firewall mechanism if certain well-formed conditions are not satisfied by loaded applications. Therefore, a bytecode verifier (BCV) formally verifies those conditions. The BCV is out of the scope of the Java Card System defined in [5].

The Java Card API (JCAPI) provides classes and interfaces for the core functionality of a Java Card application. It defines the calling conventions by which an applet may access the JCRE and services such as, among others, I/O management functions, PIN and cryptographic specific management and the exceptions mechanism. The JCAPI is compatible with formal international standards, such as ISO 7816 and industry specific standards (such as EMV, Calypso).

2.5.4 GlobalPlatform

The TOE is compliant with the GlobalPlatform 2.2.1 (GP) standard [9] which provides a set of APIs and technologies to perform in a secure way, the operations involved in the management of the applications hosted by the card. Using GP maximizes the compatibility and the opportunities of communication as it becomes the current card management standard.

The main features addressed by GP are:

- 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.

These operations are addressed by a set of APIs used by the applications hosted on the card in order to communicate with the external world on a standard basis.

The version considered in this document is version 2.2.1 of the GP Card specification. 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 support
- SCP80 support defined by the ETSI [35] (mandatory for the ISD)
- Support for contact and contactless cards (ATQ, different implicit selection on different interfaces and channels)
- Support for Supplementary Security Domains
- Installation of Security Domains
- Trusted Path privilege
- Delegated Management privilege
- Post-issuance personalization of Security Domain [12]
- Application personalization [12]
- Confidential Card Content Management [11].

The Authorized Management privilege is not supported by the TOE.

2.5.5 DESFire

The DESFire is a native application, specified by NXP and developed by Oberthur Technologies. The application has to be used with proximity Coupling Device according to ISO 1443 Type A.

DESFire is used for secure contactless transport application and related loyalty programs or access control systems.

The Desfire is part of the TSF.

2.5.6 Mifare Classic

The MIFARE classic is a proprietary technologies based upon various levels of the ISO/IEC 14443 Type A 13.56 MHz contactless smart card standard.

Its employ a proprietary protocol compliant to parts (but not all) of ISO/IEC 14443-3 Type A, with an NXP proprietary security protocol for authentication and ciphering.

The Mifare Classic does not offer any security function in the present evaluation.

2.5.7 Cipurse

The CIPURSE open security standard was established by the Open Standard for Public Transportation (OSPT) Alliance to address the needs of local and regional transit authorities for automatic fare collection systems based on smart card technologies and advanced security measures.

The CIPURSE API are developed by Oberthur Technologies. The “CIPURSE Server Crypto” API Rev 2.0 are included.

The Cipurse does not offer any security function in the present evaluation.

2.5.8 Calypso

Calypso is an international electronic ticketing standard for contactless smart cards. The dF4 includes APIs to support Calypso application Rev3.2 application developed by Calypso Network Association. The application has to be used with proximity Coupling Device according to ISO 1443 Type B following the set of specifications from the Calypso Network Association.

The Calypso does not offer any security function in the present evaluation.

2.5.9 M4M

A mobile device that implements MIFARE Technology can act as the replacement of a number of traditional plastic cards that the user of the mobile device would otherwise carry in his or her pocket. The MIFARE Implementation therefore provides in the secure element of the mobile device a number of Virtual Cards (VCs), which are largely comparable to the traditional plastic cards. From the M4M 2.0 release onwards those Virtual Cards can be of type:

- MIFARE Classic
- MIFARE DESFire

A MIFARE Implementation can generally support multiple VCs of multiple types, although there is no minimum number of VCs of either type to be supported.

The Mifare For Mobile implements several applets developed by Oberthur Technologies. These Applets allow the management of several Mifare Virtual Cards (VC) by OTA by the

owner of the UICC and by tiers parties (e.g.: providers as transport companies) who can create, delete and update VC.

The management of these applets and of these VC can be done by OTA.

The M4M APIs are included in this present evaluation and M4M applet is out of the scope of this evaluation.

2.5.10 Integrated Circuit (IC)

The IC is an Infineon dual interface component supports ISO 14443 Type B and Type A and SWP protocol.

This Security IC includes:

Extra features such as SWP interface.

Die integrity,

Monitoring of environmental parameters,

Protection mechanisms against faults,

Symmetric Cryptographic Processor for Triple-key, triple-DES and AES acceleration,

True Random Number Generator,

CRC calculation module

Memory Protection Unit,

Crypto Engine (Crypto@2304T) for RSA and ECC calculations.

The IC implements security features able to ensure:

- The confidentiality and the integrity of information processed and flowing through the device,
- The resistance of the Security IC to externals attacks such as physical tampering, environmental stress or any other attacks that could compromise the sensitive assets stored or flowing through it.

For security function IC details, please refer to [29]. General Characteristics are presented in the Table 1 below:

Different Chip ID	**SLE97CNFX1M50PE**
Flash size	1,500 Mbytes
ROM	64 Kbyte
General RAM	32 Kbytes
Crypto RAM	1152 bytes
SWP RAM	256 bytes
OTP	256 bytes
ContactLess interface	SWP

Table 5: IC General Characteristics

The IC is part of the TSF, the present evaluation is a composition of the IC.

2.5.11 Operating System (OS)

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:
   - Drivers related to the I/O, RAM, ROM, EEPROM, Flash memory if any, and any other hardware component present on the Security IC,
2) High Level:
   - Protocols and handlers to manage I/O,
   - Memory and file manager,
   - Cryptographic services and any other high level services provided by the OS.

The following crypto services are included in the OS:
3DES, AES, Pseudo Random SHA, SHA-1, SHA256, SHA384, RSA 1280, 1536, 1792 and 2048, KG 2048
The TOE uses and manipulates sensitive information without revealing any element of this information.
The OS is part of the TSF.

2.5.11.1 TOE Usage

The SIM, defined in the 3GPP standards as the Subscriber Identity Module, is a removable module within a GSM mobile equipment that contains the International Mobile Subscriber Identity (IMSI) which unambiguously identifies a subscriber. When the SIM is placed in a mobile equipment, users can register onto the GSM network. The primary function of the SIM is consequently used to authenticate the validity of a terminal when accessing the network. It also provides means to authenticate the end user and may store other subscriber-related information or applications such as SIM Toolkit applications as specified in [34] and [36].

The SIM is the MNO's property, and stores MNO's specific information. The USIM defined in the 3GPP standards as the Universal Subscriber Identity Module is an evolution of the SIM developed to ensure compliance within UMTS networks (also called 3G). This new generation of SIM especially includes improvements of mutual authentication mechanisms.

Note: In the following SIM and USIM are considered in the same way regarding security. This is why the term of (U)SIM is used to refer to SIM or USIM. This Security Target addresses the case of (U)SIM cards with embedded Java Card platforms.

This TOE is an open and multi application platform intended to propose security requirements for applications with critical assets, in particular:
- Banking and finance credit / debit smartcards, electronic purse (stored value cards) and electronic commerce, loyalties,
- Network based transaction processing such as mobile phones (GSM SIM cards), pay-TV (subscriber and pay-per-view cards), communication highways (Internet access and transaction processing),
- Transport and ticketing market (access control cards),
- DESFire application;
- Governmental cards (electronic passports, ID-cards, health-cards, driver license, etc.),
- Multimedia commerce and Intellectual Property Rights protection.

2.5.12 TOE Life Cycle

The TOE life cycle follows the description of the [5] and is part of the product life cycle, i.e. the (U)SIM card, which goes from product development to its usage by the final user. The product life cycle phases are those detailed in Figure 2. We refer to [PP0035] for a thorough description of Phases 1 to 7:

- Phases 1 and 2 compose the product development: Embedded Software (IC Dedicated Software, OS, Java Card System, (U)SIM applet, other platform components such as Card Manager, Applets DESFire, M4M,Cipurse, Caplypso, ...) and IC development.
- Phase 3 and 4 correspond to IC manufacturing and packaging, respectively. Some IC pre-personalisation steps may occur in Phase 3.
- Phase 5 concerns the embedding of software components within the IC.
- Phase 6 is dedicated to the product personalization prior final use.
- Phase 7 is the product operational phase.

The (U)SIM platform life cycle is composed of four stages:

- Development,
- Storage, pre-personalization and test,
- Personalization and test,
- Final usage.

These 4 stages map to the typical smart card life cycle phases as shown in Figure 2.

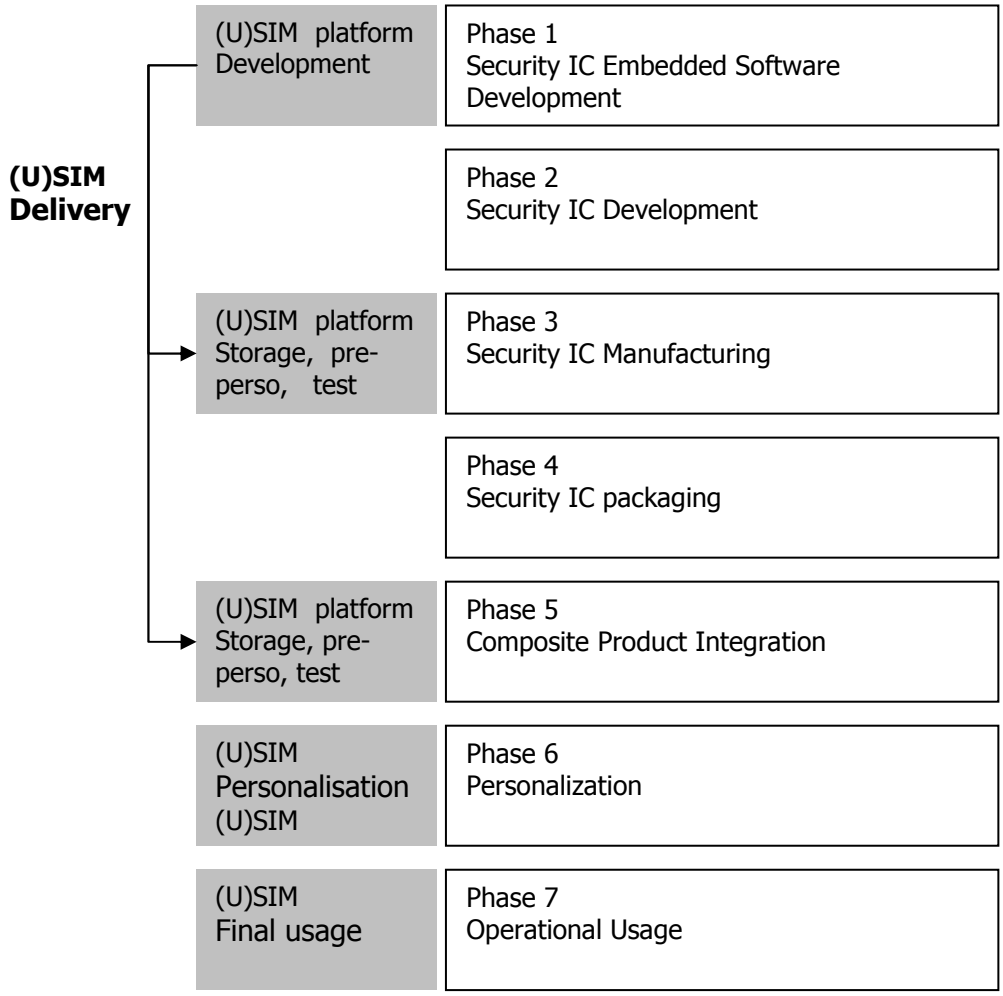


Figure 2: (U)SIM platform (TOE) Life Cycle within Product Life Cycle

(U)SIM platform Development is performed during Phase 1. This includes Java Card System (JCS) and (U)SIM conception, design, implementation, testing and documentation. The development fulfilled requirements of the final product, including conformance to Java Card Specifications, and recommendations of the SCP 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 (U)SIM platform development environment.

The delivery of the (U)SIM platform occurs in phases 5 (4, 5 and 6 are done in the same environment) at Oberthur manufacturing sites.

Delivery and acceptance procedures guaranty the authenticity, the confidentiality and integrity of the exchanged pieces. (U)SIM platform delivery involves encrypted signed sending and it supposes the previous exchange of public keys. The evaluation of the TOE includes the delivery process.

Phases 5 and 6 are done in the Oberthur manufacturing sites. They represent the code loading and personalisation of the (U)SIM Platforms. The phases take place in a controlled environment (secure locations, secure procedures and trusted personnel).

The product is tested and all critical materials including personalization data, test suites and documentation are protected from disclosure and modification.

During these phases, MNO (ISD keys and other initial data), Controlling Authority and Verification Authority data are loaded on the (U)SIM. After this phase, the (U)SIM card reaches the INITIALIZED state.

In phase 7, the (U)SIM platform provides the full set of security functionalities. These functionalities protect the product from abuse by untrusted entities.

Card management (including Secure or Basic applications loading and personalization) can occur during production in a secure area in phases 5 and 6 or during the product usage in phase 7 using an OTA bearer

All processes until the end of phase 6 are part of the assurance component ALC. (U)SIM Platform personalisation and applet loading are performed in Oberthur's industrial sites.

At the end of phase 6, the (U)SIM product is ready for use, it is sent to the customer.

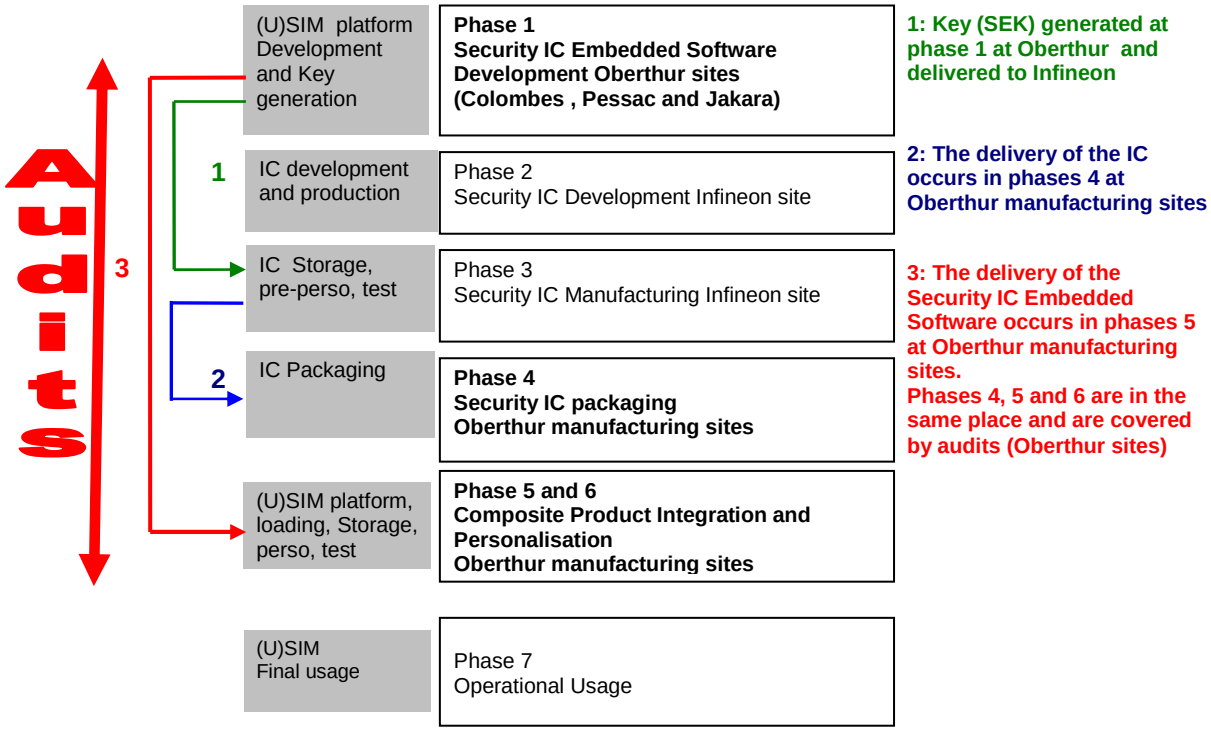


Figure 3: DragonFLY V4.0 Life Cycle within Product Life Cycle

The following steps of the life cycle are covered as specified in the table below:

Life cycle phase	Environment	Covered by
Phase 1	dragonFly Platform Development	ALC [FLY] Oberthur Sites : *Colombes and Pessac in France. Jakarta in Indonesia, Rabat* *All sites covered by an audit*
Phase 2	IC Development	ALC [IC] Infineon And [IC] EAL4+ Evaluation

Phase 3	Security IC Manufacturing	ALC [IC] Infineon Site And [IC] EAL4+ Evaluation
Phase 4	Security IC packaging TOE	ALC [FLY] Oberthur Site: Vitré and Shenzhen *Covered by an audit*
Phase 5 and 6	Construction of the TOE	ALC [FLY] Oberthur Site: Vitré and Shenzhen *Covered by an audit*
Phase 7	Operational Phase of the TOE	AGD_OPE [FLY]

[FLY] means that the audit is under the scope of this project at Oberthur promises.

[IC] is under the responsibility of Infineon. It's under the scope of IC certificate.

2.5.12.1 TOE Guidance

The table below lists the guidance for the users of the TOE.

The users of this product shall use the ST_Lite for TOE identification, details in chapter 2.3.1.	
Guidance document for development of basic application Platform	No specific recommendation for the developer of basic applications: -dragonFly v4.0 (APPLICATION DEVELOPMENT GUIDE) FQR 401 4750 Issue 1. - Guide FQR 401 4751 Issue 5 - See standard documentation for Java Card API [6], GlobalPlatform Card Specification API [9] and UICC API [66][67].
Guidance for development of secure application (application to certify on the Platform)	- See standard documentation for Java Card API [6], GlobalPlatform Card Specification API [9] and UICC API [66][67]. - dragonFly v4.0 - Application Security Recommendations - FQR 401 4749 Issue 2 - Guide FQR 401 4751 Issue 5
Guidance for development of application. Internal Guidance for Oberthur.	- 081012 11 SRS AA (APIs)
Guidance document for TOE users: Issuers, Application Providers, VA (Verification Authority), CA (Controlling Authority)	- dragonFly v4.0 - Application Management Guide FQR 401 4751 Issue 5

Guidance for Desfire	- Mifare DESFire EV1 Initialization Specification Rev.01.12 - 22. Dec 2010 NXP - Mifare DESFire EV1, Interface Specification rev. 1.0 -2008-11-21,NXP - DRAGONFLY V4.0 - DESFIRE- AGD_OPE – FQR 900 0225 Issue 5

Table 6: TOE Guidance references

Besides the listed documents, Public ST-lite is guidance available to the users of the TOE.

- **For basic applet, the platform doesn't impose any recommendation on the development of the basic applications. And the platform doesn't impose any recommendation for the Validation authority/verification authority. Except to go thru the static off card verifier (this verification is imposed by the PP USIM).**
- **This concerns basic applet to load in pre issuance and to load in post issuance.**
- **The administrator of the Desfire applications shall be trusted actors. They own or have access to the card master keys. They must keep all cryptographic keys confidential all the time.**

2.5.12.2 Loading applets

Prior to loading, the applets cap files have to follow customer recommendation if any. The applets can be loaded at phases 5, 6 and 7:

- In phases 5 and 6 the DAP mechanism is not enforced; applet loading is performed in Oberthur's industrial site, using Oberthur process.

Thus, the product can be delivered at the end of phase 6 with applets loaded onto the (U)SIM platform.

- In use phase, the DAP mechanism is mandatory.

At phase 7, (U)SIM Cards can already have some applets loaded (applets can be personalized or not personalized). Only applets with DAP signatures can be loaded to the TOE, the (U)SIM Platform in phase 7.

2.5.12.3 TOE and applications development sites and generation of the TOE

The TOE is composed of several components; all are developed at Oberthur sites. Except some component or part of -expressed in the second column- the code is a native code.

For each component, the development site is indicated in the table below.

Component/applet	**Development site**
Crypto	Colombes
BIOS, VM, JCRE, API*	Pessac
GP: Card Manager, SD, Amend A/B/C/D/E	Pessac (CM partially developed in java)
M4M API	Pessac

API Java card	Pessac
API Calypso	Pessac
API Cipurse	Rabat
DESFIRE Core	Rabat
Mifare API	Colombes
DESFIRE API	Pessac
OS Mobile	Jakarta
MifareClassic	Pessac/Infineon (for RF Library)
Telecom : USIM, SIM,UICC, perso commands	Jakarta
OTA: SCP80, RFM, RAM, BIP	Jakarta
OTA: SCP81	Pessac (partially in Java)
ETSI API : USIM Access, USIM Toolkit, NFC	Colombes

Table 7: Product component list

- Except the others API specified in the table

The integration of the components, as listed in the table above, is done at Jakarta. For this, all the code (obfuscated), without sensitive information, is made available at Jakarta.

The full generation of the final product is done at Colombes, by the Configuration Management Manager as expressed in Oberthur Technologies process. The final generated complete code to be loaded in the IC, is encrypted with the key exchanged with Infineon as expressed in Oberthur Technologies process.

These steps are covered by an audit, under the ALC_DVS.2.

2.5.13 Actors of the TOE

One of the characteristics of the (U)SIM Java Card platforms is that several entities are represented within these platforms:

- The **Mobile Network Operator** (MNO or mobile operator), issuer of the (U)SIM Java Card platform and proprietary of the TOE. The TOE guarantees that the issuer, once authenticated, can manage the loading, instantiation and deletion of applications.
- The **Application Provider** (AP), entity or institution responsible for the applications and their associated services. It could be a financial institution (a bank), a transport operator or a third party operator.
- The **Controlling Authority** (CA), entity independent from the MNO, represented on the (U)SIM card and responsible for securing the keys creation and personalization of the Application Provider Security Domain (APSD) (Push and Pull personalization model of [11]).
- The **Verification Authority** (VA), trusted third party represented on the (U)SIM card, acting on behalf of the MNO and responsible for the verification of

applications signatures (Mandated DAP) during the loading process. These applications shall be validated for the Basic ones or certified for the Secure ones.

2.5.14 TOE Security Features

Secure or Basic applets can be loaded and instantiated onto the TOE either before card issuance or over-the-air (OTA) in post-issuance through the mobile network, without physical manipulation of the TOE and in a connected environment. Besides these, other administrative operations can also be done OTA.

The main security feature of the TOE is the correct and secure execution of sensitive applications, in a connected environment and with the presence on the TOE of Basic (non-certified) applications.

The table below identify for each systems and subsystems from if it contains TSF or not.

System	Subsystem	Contains TOE Security Functional
TELECOM		
	(U)SIM	No
	Perso Cmds	No
	UICC	No
	OTA RFM	No
	OTA RAM	No
	OTA SCP 80	Yes
	OTA BIP	No
GlobalPlatform 2.2		
	GlobalPlatform (GP) API	Yes
	OpenPlatform (OP) API	Yes
	Contactless Registry Services (CRS) API	Yes
	Amd A	Yes
	Amd B (desactivated)	No
	Amd C	No
	Amd D	No
	Amd E	Yes
	GP CL Registry	Yes
JavaCard 3.04 Classic Edition		
	JCRE	Yes
	JCVM	Yes
	JCAPI	Yes

Security Domains		
	ISD	Yes
	SD1,… SDx	Yes
ETSI API		
	(U)SIM Access API	No
	(U)SIM Toolkit API	No
	NFC API	No
Operating System		
	Crypto	Yes
	Bios	Yes
DESFIRE EV1	Transport application	Yes
MIFARE Classic	Transport application	No
MIFARE 4 MOBILE	Transport application	No
CIPURSE/CALYPSO	Transport API	No
Integrated Circuit	Hardware	Yes

Table 8: Subsystems of the TOE

The TOE is composed of TOE Security Functionality (TSF) that implements security requirements and Non TOE Security Functionality (NON TSF). All are part of this evaluation (even if some don't contain TSF).

2.5.14.1 Security Services to Applications

The TOE offers to applications a panel of security services in order to protect application data and assets:

- Confidentiality and integrity of cryptographic keys and associated operations. Cryptographic operations are protected, including protection against observation or perturbation attacks. Confidentiality and integrity of cryptographic keys and application data are guaranteed at all time during execution of cryptographic operations.
- Confidentiality and integrity of authentication data. Authentication data are protected, including protection against observation or perturbation attacks. Confidentiality and integrity of authentication data and application data are guaranteed at all time during execution of authentication operations.
- Confidentiality and integrity of application data among applications. Applications belonging to different contexts are isolated from each other. Application data are not accessible by a normal or abnormal execution of another Basic or Secure application.
- Application code execution integrity. The Java Card VM and the "applications isolation" property guarantee that the application code is operating as specified in absence of perturbations. In case of perturbation, this TOE security feature must also be valid.

2.5.14.2 Application Management

The TOE offers additional security services for applications management, relying on the GlobalPlatform framework:

- The MNO as Card issuer is initially the only entity authorized to manage applications (loading, instantiation, deletion) through a secure communication channel with the card, based on SMS or BIP technology. However, the MNO can grant these privileges to the AP through the Delegated Management functionality of GP.
- Before loading, all applications are verified by a validation laboratory for the Basic applications, or by an ITSEF for the secure applications. All loaded applications are associated at load time to a Verification Authority (VA) signature (Mandated DAP) that is verified on card by the on-card representative of the VA prior to the completion of the application loading operation and prior to the instantiation of any applet defined in the loaded application.
- Application Providers personalize their applications and Security Domains (APSD) in a confidential manner. Application Providers have Security Domain key sets enabling them to be authenticated to the corresponding Security Domain and to establish a trusted channel between the TOE and an external trusted device. These Security Domains key sets are not known by the Card issuer.

Basic and Secure applets (as defined below) are loaded in different Java Card packages.

2.5.15 Non-TOE available to the TOE

2.5.15.1 Mobile Terminals

The (U)SIM as a smart card is intended to be plugged in a mobile handset. This equipment can be a mobile phone or a PDA or any other connecting device.

2.5.15.2 Basic Applets

Basic applets stand for applications that do not require any particular security for their own. This is the case for fidelity applications, Information-on-demand (IOD) applications, etc.

Platform Fly does not require any specific recommendation for basic applet.

When shared libraries are used by the Basic Application, the VA checks that the Java Card binary compatibility rules are enforced for those libraries. For each library, the VA checks that:

- The major version of the shared library found on the target platform is equal to the major version number of the library that has been used for compilation of the Basic Application.
- The minor version of the shared library found on the target platform is equal to or higher than the minor version number of the library that has been used for compilation of the Basic Application.

When the Basic Application provides shared libraries, the VA has to verify that the versioning policy is enforced for any of these libraries. Regarding previous versions of a library, the VA checks that at least the minor version changes if there are changes to the implementation of the exported methods. This rule is applicable only if the backward compatibility is ensured. For major incompatibility changes, the VA checks that major version has been increased instead.

Once an application is verified by a static verifier, it is signed and then can be loaded. The signature is checked by the Platform during the loading phase (use phase 7).

2.5.15.3 Secure Applets

Secure applets are applications requiring a high level of security for their own assets. It is indeed necessary to protect application assets in confidentiality, integrity or availability at different security levels depending on the AP Security Policy. This is typically the case for payment applications, requiring a high level of security assurance (the Common Criteria EAL4 with AVA_VAN.5 or higher EAL is required), for conditional access mobile TV applications or digital signature applications (in Europe, the PP [SSCD] is required for qualified digital signature applications).

As such, secure applications follow a Common Criteria evaluation and certification in composition with the previously certified (U)SIM smart card.

2.5.15.4 Terminals, Remote Servers and Trusted IT Products

Using its NFC (contactless) interface, the TOE can communicate with a card reader such as POS (Point of Sale) equipments or ticketing systems terminals. These terminals are responsible for the protection of their own assets.

The Platform NFC with DESFire can communicate with the ticketing systems terminals (Contactless). This ticketing systems terminals shall be conformant to:

-Mifare DESFire EV1 Initialization Specification Rev.01.12 - 22. Dec 2010 NXP,

-and Mifare DESFire EV1, Interface Specification rev. 1.0 -2008-11-21,NXP.

Using the BIP interface or SMS, the TOE can also communicate with remote servers, for instance for remote administration or transfer of applicative data. For sensitive operations, such as remote administration, the TOE may require mutual authentication or the use of secure channels. In that case, the keys and/or certificates required for these operations on the TOE will also have to be available from the remote server and protected. The remote server and, if any, the device (such a HSM) from which the keys are obtained are referred as a Trusted IT product.

2.5.15.5 Off card Verifier

The bytecode verifier is a program that performs static checks on the byte codes of the methods of a CAP file.

Bytecode verification is a ***key*** component of security: applet isolation, for instance, depends on the file satisfying the properties a verifier checks to hold. The verification made by the verifier ensures that a method of a CAP file does not contain, for instance, an instruction that allows forging a memory address or an instruction that makes improper use of a return address as if it were an object reference. In other words, bytecodes are verified to hold up to the intended use to which they are defined. This TOE considers static bytecode verification; it has to be performed on the host at *off-card* verification and prior to the installation of the file on the card in any case.

3 Conformance Claims

3.1 CC Conformance Claims

This Security Target claims conformance to **CC version 3.1** with the following documents:

[1] "Common Criteria for information Technology Security Evaluation, Part 1: Introduction and general model", September 2012, Version 3.1 revision 4.

[2] "Common Criteria for information Technology Security Evaluation, Part 2: Security Functional requirements", September 2012, Version 3.1 revision 4.

[3] "Common Criteria for information Technology Security Evaluation, Part 3: Security Assurance requirements", September 2012, Version 3.1 revision 4.

Conformance is claimed as follows:

Part 1: conformant
Part 2: extended with the FCS_RNG.1. All the other Security requirements have been drawn from the catalogue of requirements in Part 2
Part 3: conformant EAL4 augmented with ALC_DVS.2, ALC_FLR.3 and AVA_VAN.5.

3.2 ST Conformance Claims

This ST is conformant to the “Java Card System Open Configuration Protection Profile” [5] and to “(U)SIM Java Card Platform Protection Profile Basic Configuration” [30] .

3.3 Conformance Claims to PPs

This Security Target is **demonstrable** conformant to the 2 listed PPs in chapter 3.2.

3.4 Conformance Claims Rationale

What follows reveals the consistency between this ST and the 2 PPs. To avoid redundancies, only the differences between this ST and chapter 2.4 of [30] for conformity between the 2 PPs are expressed in this ST.

3.4.1 TOE Type Conformance

The TOE type is the (U)SIM Java Card Platform as expressed in the USIM PP [30] comprehends the Java Card System defined [5], including the DESFire, the OS and the IC.

3.4.2 Additional SPD (Security Problem Definition)

The Java Card System threats, OSPs and assumptions are relevant to the TOE as defined in this ST.

The (U)SIM Java card, (U)SIM threats, OSPs and assumptions are relevant to the TOE as defined in this ST.

SPD for DESFire are added to answer to be conformant to MIFARE DESFire EV.

3.4.2.1 Assets

All assets of [5] and those of [30] are included in this ST. The additional assets for DESFire are listed below:

D.DF_DATA

Keys, Files and Values controlled by the MIFARE DESFire,
To be protected from unauthorized disclosure or modification.

D.DF_CODE

MIFARE DESFire, stored and in operation.
To be protected from unauthorized disclosure or modification.

3.4.2.2 Threats

All threats of [5] and those of [30] are included in this ST.

T.DF_DATA_MODIFICATION

User data stored by the TOE may be modified by unauthorised subjects. This threat applies to the processing of modification commands received by the TOE, it is not concerned with verification of authenticity.

T.DF_IMPERSONATE

Impersonating authorized users during the authentication process of the MIFARE DESFire.

An unauthorized subject may try to impersonate an authorized subject during the authentication sequence of the MIFARE DESFire, e.g. by a man-in-the middle or replay attack.

T.DF_CLONING

Cloning using keys, files and values maintained MIFARE DESFire.

Keys, files and values maintained by the MIFARE DESFire stored on the TOE may be read out by an unauthorized subject in order to create a duplicate.

3.4.2.3 OSPs

All OSPs of [5] and those of [30] are included in this ST.

3.4.2.4 Assumptions

Assumptions of [5] and [30] are included in this ST except:

- A.Production, As Production and personalization environment of the TOE is part of the developer environment; it will be evaluated by ALC_DVS.2.

- A.Personaliser is removed, the operator that is in charge of the TOE personalization process this assumption is part of the evaluation with the ALC_DVS.2. He ensures the security of the keys he loads on the (U)SIM cards.

- A.Deletion is removed, the deletion of applets is in the scope of the evaluation as O.CARD-MANAGEMENT is an objective in this ST.

- A.Verification, all the applets bytecodes are verified at least once, before the loading, in order to ensure that each bytecode is valid at execution time. This concerns loading in pres-issuance and post-issuance applications.

For pres issuance, this assumption will be evaluated by ALC_DVS.2 .

For post issuance, 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 in order to ensure that each bytecode is valid at execution time.

The DESFire adds additional assumptions on the secure value used and terminal support:

A. DF_Secure _Value

Usage of secure values: Only confidential and secure keys shall be used to set up the authentication and access rights in DESFire. These values are generated outside the TOE and they are downloaded on the TOE.

A.DF_Terminal_Support

Terminal support to ensure integrity and confidentiality:

The terminal verifies information sent by the TOE in order to ensure integrity and confidentiality of the communication.

3.4.3 Security Objectives

All security objectives of [5] and [30] are included in this ST plus those added for the DESFire.

3.4.3.1 Security Objectives for the Operational Environment

All Security Objectives for the Environment of [5] are included in this ST, except:

-OE.CARD-MANAGEMENT, OE.SCP.SUPPORT, OE.SCP.IC, OE.SCP.RECOVERY which are in this ST a refined security objective for the TOE.

-The Security Objective OE.PRODUCTION for the Operational Environment is removed, the production is part of development environment, and this requirement, in this ST is fulfilled by ALC_DVS.2.

-The Security Objective OE.PERSONALIZER for the Operational Environment is removed, the personalization is part of development environment, and this requirement, in this ST is fulfilled by ALC_DVS.2.

-The Security Objective OE.CODE-Evidence ensuring that loaded application has not been changed since the code verifications required in OE.VERIFICATION is fulfilled by ALC_DVS.2 for pre issuance.

-The Security Objective OE.Verification ensuring that loaded application has been checked (by off card verifier) is fulfilled by ALC_DVS.2 for loading applications in pres-issuance phase.

For post issuance, this assumption is still valid, the applets are checked by the Verification Authority and signed, see guidance of the TOE. The mandated Dap mechanism (mandatory at post-issuance) implemented in the Card manager ensures that the integrity of the application is maintained after it's verification by the VA.

-The OE.SHARE-CONTROL is modified; the basic application is removed from this objective: All sensitive applications must have means to identify the applications with whom they share data using the Shareable Interface.

The basic applications are removed from this objective as when a basic application implementing a Shareable Interface wants to share its data with any basic application; there is no need to add any new validation of this basic application.

Application note:
By this sharing, if the implementation of the basic application is modified, then there is a new validation.

This objective concerns only sensitive applications. The Shareable interface functionality should be strictly controlled for all sensitive applications. So, if a sensitive application implementing a Shareable Interface has to share data with a new application (sensitive or basic) it has to be updated, and thus re-certified.

The 2 following objectives for the operational Environment concerns the DESFire:

OE.DF_Secure_Values

Generation of secure values:

The environment shall generate confidential and secure keys for authentication purpose. These values are generated outside the TOE and they are downloaded to the TOE during the personalization or usage in phase 5 to 7.

OE.DF_Terminal_Support

Terminal support to ensure integrity and confidentiality:

The terminal shall verify information sent by the TOE in order to ensure integrity and confidentiality of the communication. This involves checking of MAC values, verification of redundancy information according to the cryptographic protocol and secure closing of the communication session in phase 7.

To avoid redundancies, Additional security Objectives for the TOE to address the desfire and the SCP are described in chapters 5.1.3 and 5.1.4.

3.4.4 SFRs and SARs Statements Consistency

3.4.4.1 SFRs Consistency

The SFRs of the Java Card System and SFRs from (U)SIM are relevant to the TOE as defined in this ST. All are included in this ST.

This ST adds some SFRs identified in this ST by adding '-OT' in the end of SFR name.

The list of SFRS with justification is presented hereafter:

The platform extends initial PP requirement to allow the card management based on Token_RSA, Token_TDES, Token_AES and return the 3 different receipts:

FCS_COP.1/TOKEN-OT_RSA Cryptographic operation;

FCS_COP.1/TOKEN-OT_TDES Cryptographic operation;

FCS_COP.1/TOKEN-OT_AES Cryptographic operation;

FCS_COP.1/RECEIPTS-OT_TDS Cryptographic operation;

FCS_COP.1/RECEIPTS-OT_AES Cryptographic operation;

The platform allows confidential loading. It is an additional function to ensure the confidentiality of applet code at loading phase. This SFR is present and is not mandatory to use by the user of the TOE.

FCS_COP.1/CIPHERLOADFILE-OT Cryptographic operation;

2 Cryptographic functions are available: TDES and AES.

FCS_COP.1/CIPHERLOADFILE-OT _TDES Cryptographic operation;

FCS_COP.1/CIPHERLOADFILE-OT _AES Cryptographic operation;

The TOE adds FPT_TDC.1/CCM-OT to enforce the consistency of security attributes when transmitted to a Security Domain.

FPT_TDC.1/CCM-OT Inter-TSF basic TSF data consistency;

The 2 SFRs are added to check the result of authentication related to the opening of secure communication channel with the card and to protect the feedback.

FIA_AFL.1/SC-OT Authentication failure handling;

FIA_UAU.7/SC-OT Protected authentication feedback;

This SFR FPR_UNO.1. is added to ensure the unobservability of import and use of keys values.

FPR_UNO.1/SC-OT Unobservability;

To fulfil the 3 objectives of the OS and the IC of the TOE: O.SCP.SUPPORT, O.SCP.IC, O.SCP.RECOVERY, this ST adds additional SFRs: those from the ICs listed in the public ST of the IC [28] and the list presented hereafter:

FPT_FLS.1/SCP Failure with preservation of secure state

FRU_FLT.1/SCP Degraded fault tolerance

FPT_PHP.3/SCP Resistance to physical attack

FPT_RCV.3/SCP Automated recovery without undue loss

FPT_RCV.4/SCP Function recovery

FCS_RNG.1/SCP Random number generation

The PP (U)SIM requires that FCS_COP.1/DAP meets SSA-PKCS1 in compliance with PKCS#1-v1_5, this ST claims conformity to PKCS#1-v2_1.

Between the 2 versions 1._5 and 2.1, there is the version 2.0:

Version 2.0 incorporated major editorial changes in terms of the document structure and introduced the RSAES-OAEP encryption scheme. This version continued to support the encryption and signature processes in version 1.5, although the hash algorithm MD4 was no longer allowed due to cryptanalytic advances in the intervening years. Version 2.0 was republished as IETF RFC 2437 [35].

Version 2.1 introduces multi-prime RSA and the RSASSA-PSS signature scheme with appendix along with several editorial improvements. This version continues to support the schemes in version 2.0.

So PKCS#1-v2_1 is compatible with PKCS#1-v1_5.

3.4.4.2 DESFire additional SFRs

To fulfil the DESFire objectives, additional SFRs are added they are listed here after:

All SFRs details are in chapter 7.1.3.

The added SFRs **(FMT_SMR.1 ,FDP_ACC.1,FDP_ACF.1, FMT_MSA.3, FMT_MSA.1; FMT_SMF.1, FDP_ITC.2, FCS_CKM.4,FMT_MTD.1)** ensure the access control to the DESFire application, and doesn't concern the other access control of the TOE.

The added SFRs (with the use of Crypto SFRS-FCS_COP.1) ensure the authentication and its protection of the users of the DESFire.

FIA_UID.2
FIA_UAU.2
FIA_UAU.5
FTP_TRP.1
FPT_RPL.1

The authentication of the users within the DESFire is ensured by**: FTP_TRP.1 & FPT_RPL.1** In addition to **FCS_COP.1** already implemented in the TOE.

The consistency of the data exchanged between the DeESFire and the terminal is ensured by **FPT_TDC.1.**
The FDP_ROL.1 ensures the possibility to rollback a set of modifying operations on backup files in total.

3.4.4.3 SFRs additional refinements and precisions

In this ST, additional operation is added to FMT_MSA.1/SC.

FMT_MSA.1.1: The TSF shall enforce the Secure Channel Protocol (SCP) information flow control policy to restrict the ability to **delete and modify** the security attributes AS.KEYSET_VERSION and AS.KEYSET_VALUE to Security Domain.

In the PP USIM [30], the operation is restricted to **Modify**.

In the PP USIM [30], the SFR FCS_COP.1/DAP is defined for 2 algorithms: PKC Scheme **or** DES Scheme. The TOE implements only the PKC Scheme: SHA-1 hash and PKCS#1 RSA signature.

FCS_COP.1.1/DAP_1: The TSF shall perform verification of the DAP signature attached to Executable Load Applications in accordance with a specified cryptographic algorithm

- PKC Scheme: SHA-1 hash and PKCS#1 RSA signature

and cryptographic key sizes

- PKC Scheme: RSA key length 1024 bits

that meet the following:

- Sections C.1.2 and C.6 of [9]
- PKC Scheme: SSA-PKCS1-v1_5 as defined in PKCS#1

FCS_COP.1.1/DAP_2: The TSF shall perform verification of the DAP signature attached to Executable Load Applications in accordance with a specified cryptographic algorithm

- PKC Scheme: SHA-1 hash and PKCS#1 RSA signature

and cryptographic key sizes

- PKC Scheme: RSA key length 2048 bits

that meet the following:

- Sections C.1.2 and C.6 of [9]
- PKC Scheme: SSA-PKCS1-v1_5 as defined in PKCS#1

FMT_SMF.1/SC: The TSF shall be capable of performing the following management functions:

- Management functions specified in GlobalPlatform specifications [9]:
  - loading (Section 9.3.5 of [9]);
  - installation (Section 9.3.6 of [9]);
  - extradition (Section 9.4.1 of [9]);
  - registry update (Section 9.4.2 of [9]);
  - deletion (Section 9.5 of [9]);

- SD personalization rules, Pull and Push model (Section 11 of [12]).

This security target adds also additional refinement to:

FMT_MSA.1/SD is refined against [USIM_PP] (a query operation is added).

> Before modification of the security attributes of the Card life Cycle state, the Security Domain Application Instance is allowed to know it before modifying it.

3.4.4.4 SARs Additional refinement

This ST adds additional refinement for ALC_DVS.2 [see chapter 7.2.3.4 ALC_DVS Development security] it takes into account the personalization phase which is part of the development environment.

3.4.4.5 SARs Consistency

This ST (EAL4 augmented with ALC_DVS.2, ALC_FLR.3 and AVA_VAN.5) adds ALC_FLR.3 component to common set of the 2 PPs assurance (EAL4 augmented with ALC_DVS.2 and AVA_VAN.5).

4 Security problem definition

4.1 Assets

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

They are divided first following the three parts. The first part is related to USIM, the second part to Java Card and the third is related to the DESFire. Each is also divided in 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). For each asset it is specified the kind of risks they run.

4.1.1 USIM TOE

This section describes the assets for the (U)SIM.

4.1.1.1 User Data

The following assets specialize the asset D.APP_KEYs from [5].

D.APSD_KEYS

These keys are Application Provider Security Domains cryptographic keys needed to establish secure channels with the AP. These keys can be used to load and install applications on the card if the Security Domain has the appropriate privileges.

To be protected from unauthorized disclosure and modification.

D.CASD_KEYS

These keys are Controlling Authority Security Domains cryptographic keys needed to establish secure channels with the CA and to decrypt confidential content for APSDs.

To be protected from unauthorized disclosure and modification.

D.ISD_KEYS

These keys are Issuer Security Domain cryptographic keys needed to perform card management operations on the card.

To be protected from unauthorized disclosure and modification.

D.VASD_KEYS

These keys are Verification Authority Security Domain cryptographic keys needed to verify applications Mandated DAP signature.

To be protected from unauthorized disclosure and modification.

D.(U)SIM_DATA

Private data of the (U)SIM application, like the contents of its private fields.

To be protected from unauthorized disclosure and modification.

D.(U)SIM_CODE

The code of the (U)SIM application on the card.

To be protected from unauthorized modification.

4.1.1.2 TSF Data

D.GP_CODE

The code of the GlobalPlatform framework on the card.

To be protected from unauthorized modification.

D.CARD_MNGT_DATA

The data of the card management, like for instance, the identifiers, the privileges, life cycle states, the memory resource quotas of applets and security domains.

To be protected from unauthorized modification.

4.1.2 Java Card System Protection Profile - Open Configuration

This part is related to Java card assets.

4.1.2.1 User data

D.APP_CODE

The code of the applets and libraries loaded on the card.

To be protected from unauthorized modification.

D.APP_C_DATA

Confidential sensitive data of the applications, like the data contained in an object, a static field of a package, a local variable of the currently executed method, or a position of the operand stack.

To be protected from unauthorized disclosure.

D.APP_I_DATA

Integrity sensitive data of the applications, like the data contained in an object, a static field of a package, a local variable of the currently executed method, or a position of the operand stack.

To be protected from unauthorized modification.

D.APP_KEYs

Cryptographic keys owned by the applets.

To be protected from unauthorized disclosure and modification.

D.PIN

Any end-user's PIN.

To be protected from unauthorized disclosure and modification.

4.1.2.2 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.

4.1.3 DESFire

This section describes the assets for the DESFire as specified by NXP.

Mifare DESFire Operating System Data, as initialisation data, configuration data, cryptographic keys, random numbers for key generation and used data by the Mifare DESFire to execute its security functions.

D.DF_DATA

Keys, Files and Values controlled by the MIFARE DESFire,

To be protected from unauthorized disclosure or modification.

D.DF_CODE

MIFARE DESFire, stored and in operation.

To be protected from unauthorized modification.

4.2 Users / Subjects

Subjects are active components of the TOE that (essentially) act on the behalf of users. Users of the TOE include people or institutions (like the AP, the MNO and the VA), hardware (like the CAD where the card is inserted) and software components (like the application packages installed on the card).

In this Security target, relevant subjects are those listed in [5] plus the following ones:

4.2.1 USIM TOE

This section describes the subjects for the (U)SIM part.

S.SD

A GlobalPlatform Security Domain representing on the card a off-card entity. This entity can be the Issuer, an Application Provider, the Controlling Authority or the Validation Authority.

4.2.2 DESFIRE

The users/subjects considered for the Desfire are specified in the security Target Lite [75]

4.3 Threats

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

The first group of threats is taken from the PP [30](U)SIM Platform chapter 4.3.1. The second group in 4.3.2 is from Java card PP [5] and the third is related to DESFire.

4.3.1 (U)SIM Part

This section describes threats for the (U)SIM part.

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, SPA 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 listed in this ST, including all assets related to the TOE code : **D.DF_CODE, D.(U)SIM_CODE** ,**D.GP_CODE**.

T.INTEG-USER-DATA

The attacker through a malicious applet loaded on the card modifies application data, application keys or authentication data.

Directly threatened asset(s): **D.APP_I_DATA**, **D.(U)SIM_API_DATA**, **D.APSD_KEYS** and **D.CASD_KEYS, D.ISD_KEYS, D.VASD_KEYS**.

T.COM_EXPLOIT

An attacker remotely exploits the communication channel (ISO-7816, NFC, BIP or SMS) established between the mobile phone and the (U)SIM card in order to modify or disclose confidential data.

All assets are threatened.

T.UNAUTHORIZED_CARD_MNGT

The attacker performs unauthorized card management operations (for instance impersonates one of the actor represented on the card) in order to take benefit of the privileges or services granted to this actor on the card such as fraudulent:

- load of a package file
- installation of a package file
- extradition of a package file or an applet
- personalization of an applet or a Security Domain
- deletion of a package file or an applet
- privileges update of an applet or a Security Domain

Directly threatened asset(s): **D.ISD_KEYS**, **D.CASD_KEYS**, **D.APSD_KEYS**, **D.VASD_KEYS**, **D.APP_C_DATA**, **D.APP_I_DATA**, **D.APP_CODE** and **D.CARD_MNGT_DATA**.

T.LIFE_CYCLE

An attacker accesses to an application outside of its expected availability range thus violating irreversible life cycle phases of the application (for instance, an attacker re-personalizes the application).

Directly threatened asset(s): **D.APP_I_DATA**, **D.APP_C_DATA**, and **D.CARD_MNGT_DATA**.

T.UNAUTHORIZED_ACCESS

By using the shareable object mechanism on which relies the communication between two applets, the attacker uses an applet on card to get access or to modify data from another applet that he should not have access to.

All assets are threatened.

4.3.2 Java Card System Protection Profile Open Configuration

This section introduces the threats to the assets against which specific protection within the TOE or its environment is required.

4.3.2.1 CONFIDENTIALITY

T.CONFID-APPLI-DATA

The attacker executes an application to disclose data belonging to another application.

Directly threatened asset(s): **D.APP_C_DATA**, **D.PIN** and **D.APP_KEYs**.

T.CONFID-JCS-CODE

The attacker executes an application to disclose the Java Card System code.

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.

Directly threatened asset(s): **D.API_DATA, D.SEC_DATA, D.JCS_DATA** and **D.CRYPTO**.

4.3.2.2 INTEGRITY

T.INTEG-APPLI-CODE

The attacker executes an application to alter (part of) its own code or another application's code.

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.

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

T.INTEG-APPLI-DATA

The attacker executes an application to alter (part of) another application's data.

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.

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.

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.

Directly threatened asset(s): **D.API_DATA**, **D.SEC_DATA**, **D.JCS_DATA** and **D.CRYPTO**.

Other attacks are in general related to one of the above, and aimed at disclosing or modifying on-card information. Nevertheless, they vary greatly on the employed means and threatened assets, and are thus covered by quite different objectives in the sequel. That is why a more detailed list is given hereafter.

4.3.2.3 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.

Directly threatened asset(s): **D.SEC_DATA** (other assets may be jeopardized should this attack succeed, for instance, if the identity of the JCRE is usurped), **D.PIN** and **D.APP_KEYs**.

T.SID.2

The attacker modifies the TOE's attribution of a privileged role (e.g. default applet and currently selected applet), which allows illegal impersonation of this role.

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

4.3.2.4 UNAUTHORIZED EXECUTION

T.EXE-CODE.1

An applet performs an unauthorized execution of a method.

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

T.EXE-CODE.2

An applet performs an execution of a method fragment or arbitrary data.

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.

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

T.EXE-CODE-REMOTE

The attacker performs an unauthorized remote execution of a method from the CAD. Directly threatened asset(s): D.APP_CODE.

4.3.2.5 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. Directly threatened asset(s): D.JCS_DATA.

4.3.2.6 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.

Directly threatened asset(s): D.APP_C_DATA, D.APP_I_DATA and D.APP_KEYs.

4.3.2.7 CARD MANAGEMENT

T.INSTALL

The attacker fraudulently installs post-issuance 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.

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).

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).

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

4.3.3 DESFire Part

For the DESFire, this ST includes threats related to DESFire code and Data.

Desfire assets to protect are defined in 4.1.3.

T.DF_DATA_MODIFICATION

User data stored by the TOE may be modified by unauthorised subjects. This threat applies to the processing of modification commands received by the TOE, it is not concerned with verification of authenticity.

T.DF_IMPERSONATE

Impersonating authorized users during the authentication process of the MIFARE DESFire.

An unauthorized subject may try to impersonate an authorized subject during the authentication sequence of the MIFARE DESFire, e.g. by a man-in-the middle or replay attack.

T.DF_CLONING

Cloning using keys, files and values maintained MIFARE DESFire.

Keys, files and values maintained by the MIFARE DESFire stored on the TOE may be read out by an unauthorized subject in order to create a duplicate.

4.4 Organisational Security Policies

This section describes the organizational security policies to be enforced with respect to the TOE environment. Rules, to which both the TOE and its human environment apply to, shall comply with security needs related to (U)SIM Java Card Platform.

All the OSPs listed in [5] and In the IC [29] are relevant for this Security Target. This Security Target adds the following OSPs for (U)SIM Part.

4.4.1 (U)SIM part

This section describes OSPs for the (U)SIM part.

4.4.1.1 Basic and Secure Applications Policies

This Security Target distinguishes basic from secure applets. The former must go through a validation process before being authorized to be load on the card. The latter are certified in composition with the current TOE and keep their certification independently of the other applets loaded on the card compliant with the following OSPs. Basic and Secure applets are loaded in different Java Card packages.

The platform implements the certification requirements for “open and Isolating Platform” as defined in the NOTE 10 [44].

OSP.SECURE-APPS-CERTIFICATION

Secure applications must be certified according to the Common Criteria at an EAL equal to the one of this Security Target.

These applications are associated to a digital signature which will be checked by a VA during the loading into the TOE.

Application note:

This composition process requires that platform administrator and user guides (AGD_ADM and AGD_USR) are available to the secure application developer. The Evaluation report for the composition (ETR-COMP), delivered by the ITSEF which manages applications composition, must be also provided.

OSP.BASIC-APPS-VALIDATION

Basic applications shall be associated to a digital signature which will be checked by a VA during the loading into the TOE.

In addition to the rules stated by the Java Card specification, the validation process must enforce that basic applications:

- follow the extra-rules stated in the user manual of the considered (U)SIM Java Card Platform,
- cannot be libraries,
- don’t use RMI,
- don’t use proprietary libraries which are not certified (except system libraries),
- access control to certified proprietary libraries is controlled by the secure application which has defined the library,
- are associated to an identifier and this identifier has to be used in parameter of the function calls.

Application note:

For the TOE, the restrictions for proprietary and system libraries are reported on secured applications (In particular for Shareable interfaces).
The evaluation will show that the use of libraries by basic applications have no impact on the security of the platform and/or on the secured applications.

OSP.SHARE-CONTROL

The Shareable interface functionality should be strictly controlled for all sensitive applications to prevent transitive data flows between applets (i.e., no resharing of a shareable object with a third applet) and thus prevent access to unauthorized data.

OSP.AID-MANAGEMENT

When loading an application that uses shareable object interface, to make its services available to other applications, the VA or the MNO shall verify that the AID of the application being loaded does not impersonate the AID known by another application on the card for the use of shareable services.

4.4.1.2 Loading Policies

OSP.OTA-LOADING

Application code, validated or certified depending on the application, is loaded "Over The Air" (OTA) onto (U)SIM Platform using OTA servers of the mobile operator.

If needed, for delegated management, the Card issuer can pre-authorize content loading operation through privilege to individual on-card representative of APs. In that case the application code is loaded in the APSD.

Once loaded, the application is personalized using the appropriate SD keys.

OSP.OTA-SERVERS

A security policy shall be employed by the mobile operator to ensure the security of the applications stored on its servers.

Application note:

The integrity of sensitive applications is checked during loading Phase, covered by DAP mechanism.

The confidentiality of the sensitive applets between application developer delivery and loading must be ensured:

– Confidentiality between sensitive application developer and OTA servers

– Confidentiality of sensitive applications when stored in the OTA servers,

4.4.1.3 Key Policies

OSP.APSD-KEYS

The APSD keys personalization can rely either on the key escrow if the APSD has been created before the usage phase of the (U)SIM card or on the CA if the APSD has been created during the usage phase.

In the first case, the security domain keys of the AP (APSD keys) are generated and stored in a secure way by the personalizer. Then, these keys are transmitted to the AP, via the key escrow, at the only mobile operator request.

In the second case, the APSD keys are:

- Either generated and stored in secure way by the APSD. Then these keys are securely transmitted to the AP using the CASD (Pull Model of [11]),
- Or created by the AP and securely transferred to the APSD using the CASD (Push Model of [11]).

Generated keys must be unpredictable with use of an appropriate random source used in combination with appropriate pseudo-random techniques. Compromising the security of the key generation method shall require at least as many operations as determining the value of the generated key.

Application note:

For more details concerning this OSP, refer to [11].

OSP.OPERATOR-KEYS

The security of the mobile operator keys (ISD keys) must be ensured by a well defined security policy that covers generation, storage, distribution, destruction and recovery. This policy is enforced by the mobile operator in collaboration with the personalizer.

Application note:

Token keys used to verify the tokens included in Delegated Management commands (that embed the signature of these commands) must be different for each (U)SIM card in usage.

OSP.KEY-GENERATION

The Personalizer must enforce a policy ensuring that generated keys cannot be accessed in plaintext.

Application note:

This can be applied by encrypting the generated key just after its generation with the public key of the recipient.

OSP.CASD-KEYS

The security domain keys of the CA must be securely generated and stored in the (U)SIM card during the personalization process. These keys are not modifiable after card issuance.

OSP.VASD-KEYS

The security domain keys of the VA must be securely generated and stored in the (U)SIM card during the personalization process. These keys are not modifiable after personalization of the VASD.

4.4.1.4 Platform

OSP.KEY-CHANGE

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

4.4.1.5 GlobalPlatform

OSP.SECURITY-DOMAINS

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

OSP.QUOTAS

Security domains are subject to quotas of memory at creation.

4.4.2 Java Card System Protection Profile - Open Configuration

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

OSP.VERIFICATION

This policy shall ensure the consistency between the export files used in the verification and those used for installing the verified file. The policy must also ensure that no

modification of the file is performed in between its verification and the signing by the verification authority.

Application Note:

The application bytecode verification is described in the chapter 4.4 #.VERIFICATION [5].

4.4.3 OSP for DESFIRE

OSP.DF_Confidentiality Confidentiality during communication

The TOE shall provide the possibility to protect selected data elements from eavesdropping during contact-less communication. The TOE shall also provide the possibility to detect replay or man-in-the-middle attacks within a session.

OSP.DF_Transaction Transaction mechanism

The TOE shall provide the possibility to combine a number of data modification operations in one transaction, so that either all operations or no operation at all is performed.

4.5 Assumptions

The following assumption concerns the product operational environment, after product delivery. Except assumptions listed in §3.4.2.4; those mentioned in the [5] and [30] Protection Profiles are included in this ST.

Are also added assumptions from the certified IC [29].

4.5.1 Actors

A.MOBILE-OPERATOR

The mobile operator is a trusted actor responsible for the mobile network and the associated OTA servers.

The mobile operator as Card issuer cannot get access or change the application data which belongs to the AP.

Application Note:

The integrity of sensitive application is ensured by the DAP mechanism, the confidentiality of sensitive application is ensured by OSP.OTA-SERVERS.

A.OTA-ADMIN

Administrators of the mobile operator OTA servers are trusted people. They are trained to use and administrate securely those servers. They have the means and the equipments to perform their tasks.

They are aware of the sensitivity of the assets they managed and the responsibilities associated to the administration of OTA servers.

Application note:

OTA servers' security guidance document with regular site inspections shall be employed to check the applicability of the rules.

A.APPS-PROVIDER

The AP is a trusted actor that provides basic or secure applications. He is responsible for his security domain keys (APSD keys).

Application note:

An AP generally refers to the entity that issues the application. For instance it can be a financial institution for a payment application such as EMV.

A.VERIFICATION-AUTHORITY

The VA is a trusted actor who is able to guarantee and check the digital signature attached to a basic or secure application.

Application note:

As a consequence, it guarantees the success of the application validation or certification upon loading.

A.KEY-ESCROW

The key escrow is a trusted actor in charge of the secure storage of the initial AP keys generated by the TOE personalizer during initial personalization.

A.CONTROLLING-AUTHORITY

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

4.5.2 Java Card System Protection Profile - Open Configuration

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

A.APPLET

Applets loaded post-issuance do not contain native methods. The Java Card specification explicitly "does not include support for native methods" outside the API.

A.VERIFICATION

All the bytecodes are verified at least once, before the loading, in order to ensure that each bytecode is valid at execution time.

This verification applies for applet loading in post-issuance phase, in pre-issuance, this verification is ensured by ALC_DVS.2.

4.5.3 DESFire Assumptions

A.DF_Secure _Value

Usage of secure values: Only confidential and secure keys shall be used to set up the authentication and access rights in DESFire. These values are generated outside the TOE and they are downloaded on the TOE.

A.DF_Terminal_Support

Terminal support to ensure integrity and confidentiality:

The terminal verifies information sent by the TOE in order to ensure integrity and confidentiality of the communication.

5 Security Objectives

5.1 Security Objectives for the TOE

The security objectives of the TOE comprise the security objectives given in [5], for the Java Card System, the security objectives given for the card management [30] and the security objectives given in the DESFire 76]

5.1.1 (U)SIM part

This section describes the security objectives for the (U)SIM part[30] .

5.1.1.1 Card Management

O.CARD-MANAGEMENT

The TOE shall provide card management functionalities (loading, installation, extradition, deletion of applications and GP registry updates) in charge of the life cycle of the whole (U)SIM card and installed applications (applets)

The card manager, the application with specific rights responsible for the administration of the smart card, shall control the access to card management functions. It shall also implement the card issuer's policy on card management.

Application note:

The card manager will be tightly connected in practice with the rest of the TOE, which in return shall very likely rely on the card manager for the effective enforcement of some of its security functions.

The mechanism used to ensure authentication of the TOE issuer, that manages the TOE, or of the Service Providers owning a Security Domain with card management privileges is a secure channel. This channel will be used afterwards to protect commands exchanged with the TOE in confidentiality and integrity.

The platform guarantees that only the ISD or the Service Providers owning a Security Domain with the appropriate privilege (Delegated Management) can manage the applications on the card associated with its Security Domain. This is done accordingly with the card issuer's policy on card management.

The actor performing the operation must beforehand authenticate with the Security Domain. In the case of Delegated Management, the card management command will be associated with an electronic signature (GlobalPlatform token) verified by the ISD before execution.

O.DOMAIN-RIGHTS

The Card issuer shall not get access or change personalized AP security domain keys which belong to the AP. Modification of a security domain key set is restricted to the AP who owns the security domain.

Application note:

APs have a set of keys that allows them to establish a secure channel between them and the platform. These key sets are not known by the TOE issuer. The security domain initial keys are changed before any operation on the SD (OE.KEY-CHANGE) through standard PUT KEY procedures (if the initial keys were kept by key escrow) or through one of the SD personalization mechanisms described in Section 4.3.3 of [11].

O.APPLI-AUTH

The card manager shall enforce the application security policies established by the card issuer by requiring application authentication during application loading on the card.

Application note:

Each application loaded onto the TOE has been signed by the VA. The VA will guarantee that the security policies established by the card issuer on applications are enforced. This authority is present on the TOE as a Security Domain whose role is to verify each signature at application loading.

The platform provides important extra features about application management and especially loading:

- Loaded applications are previously validated by an accredited laboratory for basic applications and certified by an accredited ITSEF for secure applications.
- All loaded applications are associated to a DAP signature generated by a VA which is verified at loading by the third party representative present on the platform (Mandated DAP verification).

5.1.1.2 Communication

O.COMM_AUTH

The TOE shall authenticate the origin of the card management requests received by the card, and authenticate itself to the remote actor.

O.COMM_INTEGRITY

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

O.COMM_CONFIDENTIALITY

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

5.1.2 Java Card System Protection Profile - Open Configuration

This section defines the security objectives to be achieved by the TOE and extracted from [5].

5.1.2.1 IDENTIFICATION

O.SID

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

5.1.2.2 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.

O.GLOBAL_ARRAYS_CONFID

The TOE shall ensure that the APDU buffer that is shared by all applications is always cleaned upon applet selection.

The TOE shall ensure that the global byte array used for the invocation of the install method of the selected applet is always cleaned after the return from the install method.

O.GLOBAL_ARRAYS_INTEG

The TOE shall ensure that only the currently selected applications may have a write access to the APDU buffer and the global byte array used for the invocation of the install method of the selected applet.

O.NATIVE

The only means that the Java Card VM shall provide for an application to execute native code is the invocation of a method of the Java Card API, or any additional API.

O.OPERATE

The TOE must ensure continued correct operation of its security functions.

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.

5.1.2.3 SERVICES

O.ALARM

The TOE shall provide appropriate feedback information upon detection of a potential security violation.

O.CIPHER

The TOE shall provide 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.

O.KEY-MNGT

The TOE shall provide means to securely manage cryptographic keys. This concerns the correct generation, distribution, access and destruction of cryptographic keys.

O.PIN-MNGT

The TOE shall provide means to securely manage PIN objects.

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 means to execute a set of operations atomically.

O.REMOTE

The TOE shall provide restricted remote access from the CAD to the services implemented by the applets on the card.

O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION and O.CIPHER are actually provided to applets in the form of Java Card APIs.

5.1.2.4 OBJECT DELETION

O.OBJ-DELETION

The TOE shall ensure the object deletion shall not break references to objects.

5.1.2.5 APPLET MANAGEMENT

O.DELETION

The TOE shall ensure that both applet and package deletion perform as expected.

O.LOAD

The TOE shall ensure that the loading of a package into the card is safe.

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.

5.1.3 DESFire part

O.DF_Access-Control Access Control to DESFire Data

The TOE must provide an access control mechanism for data stored by the MiFARE DESFire EV1. The access control mechanism shall apply to read, modify, create and delete operations for data elements and to read modify security attributes as authentication data. It shall be possible to limit the right to perform a specific operation to a specific user. The security attributes (keys) used for authentication shall never be output.

O.DF_Authentication Authentication

The MIFARE DESFire as part of the TOE must provide an authentication mechanism in order to be able to authenticate authorised users. The authentication mechanism shall be limited to the MIFARE DESFire and shall be resistant against replay or man-in-the-middle attacks within a session.

O.DF_Confidentiality Confidential Communication

The TOE must be able to protect the communication by encryption. This shall be implemented by security attributes of the DeESFire data element that enforce encrypted communication of the MIFARE DESFire for the respective data element.

During DESFire operation, the TOE shall also provide the possibility to detect replay and man-in-the-middle attacks. This shall be implemented by checking verification data to the terminal.

O.DF_Type_Consistency

The TOE must provide a consistent handling of the data types (files and values) of the DESFire EV1. This comprises over and underflow checking for values, for data file sizes and for record handling.

O.DF_Transaction

The TOE must be able to provide a transaction mechanism that allows to update multiple data elements of the MIFARE DESFire either all in common or none of them.

5.1.4 Additional Objectives on the TOE

The TOE shall support also the three following Objectives: O.SCP.SUPPORT O.SCPP.IC and O.SCP.RECOVERY

O.SCP.SUPPORT

The TOE OS shall support the following functionalities:

- (1) 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.
- (2) It provides secure low-level cryptographic processing to the Java Card System and GlobalPlatform.
- (3) 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.
- (4) 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).

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.

5.2 Security objectives for the Operational Environment

This section introduces the security objectives to be achieved by the environment associated to the TOE. The significant security objectives for the environment of the TOE are the ones linked to relevant assumptions and OSPs.

This ST includes also the security objectives of the JavaCard except:

- the card management security objective which is now part of the TOE.
- the security objectives for the IC and the OS (O.SCP-IC and O.SCP-SUPPORT respectively) which are now part of the TOE.

5.2.1 (U)SIM part

This section describes the security objectives for the operational environment for the TOE, extracted from PP [30] .

5.2.1.1 Actors

OE.MOBILE-OPERATOR

The mobile operator shall be a trusted actor responsible for the mobile network and the associated OTA servers.

OE.OTA-ADMIN

Administrators of the mobile operator OTA servers shall be trusted people. They shall be trained to use and administrate those servers. They have the means and the equipments to perform their tasks.

They must be aware of the sensitivity of the assets they manage and the responsibilities associated to the administration of OTA servers.

Application note:

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

OE.APPS-PROVIDER

The AP shall be a trusted actor that provides basic or secure application. He must be responsible of his security domain keys.

OE.VERIFICATION-AUTHORITY

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

OE.KEY-ESCROW

The key escrow shall be a trusted actor in charge of the secure storage of the AP initial keys generated by the personalizer.

OE.CONTROLLING-AUTHORITY

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

5.2.1.2 Policies

Validation and Certification

OE.SECURE-APPS-CERTIFICATION

Secure applications must be evaluated and certified at a security level higher or equal than the one of the current Protection Profile.

OE.BASIC-APPS-VALIDATION

Basic applications must be analyzed during the validation process in order to ensure that the rules for correct usage of the TOE are still enforced.

OE.AID-MANAGEMENT

The VA or the MNO shall verify that the AID of the application being loaded does not impersonate the AID known by another application on the card for the use of shareable services.

Loading

OE.OTA-LOADING

Application code, validated or certified depending on the application, is loaded "Over The Air" (OTA) onto (U)SIM Platform using OTA servers. This process should protect the confidentiality and the integrity of the loaded application code.

Application Note:

The integrity of the Application code when loaded ‘OTA’ is ensured when the applet is loaded OTA.

The confidentiality of the application code is not mandatory, it can be ensured. This requirement depends of application Provider, Issuer and application requirement.

OE.OTA-SERVERS

The mobile operator must enforce a policy to ensure the security of the applications stored on its servers.

Keys

OE.AP-KEYS

The SD keys personalizer, the AP and the key escrow must enforce a security policy on SD keys in order to secure their transmission.

Application Note:

The SD keys personalizer, the AP and the key escrow must enforce a security policy on SD keys in order to secure their transmission between the Key escrow, AP and SD Keys personalizer.

OE.OPERATOR-KEYS

The security of the mobile operator keys must be ensured in the environment of the TOE.

OE.KEY-GENERATION

The personalizer must ensure that the generated keys cannot be accessed by unauthorized users.

OE.CA-KEYS

The security domain keys of the CA must be securely generated prior storage in the (U)SIM card.

OE.VA-KEYS

The security domain keys of the VA must be securely generated prior to storage in the (U)SIM card.

5.2.1.3 Platform

OE.KEY-CHANGE

The AP must change its security domain initial keys before any operation on it.

5.2.1.4 GlobalPlatform

OE.SECURITY-DOMAINS

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

OE.QUOTAS

Security domains are subject to quotas of memory at creation.

5.2.1.5 Applications

OE.SHARE-CONTROL

All sensitive applications must have means to identify the applications with whom they share data using the Shareable Interface.

5.2.2 Java Card System Protection Profile - Open Configuration

This section introduces the security objectives to be achieved by the environment, these security objectives are extracted from PP[5].

OE.APPLET

No applet loaded post-issuance shall contain native methods.

OE.VERIFICATION

All the bytecodes shall be verified at least once, before the loading in order to ensure that each bytecode is valid at execution time.

OE.CODE-EVIDENCE

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 present evaluation.

5.2.3 DESFire

The 2 following objectives for the operational environment are linked to DESFire requirement.

OE.DF_Secure_Values

Generation of secure values:

The environment shall generate confidential and secure keys for authentication purpose. These values are generated outside the TOE and they are downloaded to the TOE during the personalization or usage in phase 5 to 7.

OE.DF_Terminal_Support_DF

Terminal support to ensure integrity and confidentiality:

The terminal shall verify information sent by the TOE in order to ensure integrity and confidentiality of the communication. This involves checking of MAC values, verification of redundancy information according to the cryptographic protocol and secure closing of the communication session in phase 7.

5.3 Security Objectives Rationale

5.3.1 Threats

5.3.1.1 (U)SIM Part

T.PHYSICAL This threat is countered by physical protections which rely on the underlying platform.

The security objectives O.SCP.SUPPORT and O.SCP.IC protect sensitive assets of the platform against loss of integrity and confidentiality and especially ensure the TSFs cannot be bypassed or altered.

T.INTEG-USER-DATA The security objective O.SCP-SUPPORT provides functionality to ensure atomicity of sensitive operations, secure low level access control and protection against bypassing of the security features of the TOE. In particular, it explicitly ensures the independent protection in integrity of the platform data.

The security objectives O.DOMAIN-RIGHTS, OE.CA-KEYS,OE.VA-KEYS and OE.AP-KEYS ensure that personalization of the application by its associated security domain is only performed by the authorized AP.

The security objectives from [5] covering the threat T.INTEG-APPLI-DATA also cover this threat.

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

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

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

- O.CARD-MANAGEMENT controls the access to card management functions such as the loading, installation, extradition or deletion of applets.
- O.COMM_AUTH prevents unauthorized users from initiating a malicious card management operation.
- O.COMM_INTEGRITY protects the integrity of the card management data while it is in transit to the (U)SIM card.
- O.APPLI-AUTH which requires for loading all applications to be authenticated.
- O.DOMAIN-RIGHTS which restricts the modification of an AP security domain keyset to the AP that owns it.

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

- O.CARD-MANAGEMENT that controls the access to card management functions such as the loading, installation, extradition or deletion of applets and prevent attacks intended to modify or exploit the current life cycle of applications
- O.DOMAIN-RIGHTS that restricts the use of an AP security domain keysets, and thus the management of the applications related to this SD, to the AP that owns it.

T.UNAUTHORIZED_ACCESS This threat is covered by the security objective on the operational environment of the TOE OE.SHARE-CONTROL which ensures that sharing objects functionality is strictly controlled to stop data transitive flows between applets and thus stop access to unauthorized data.

5.3.1.2 Java Card System Protection Profile Open Configuration

CONFIDENTIALITY

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). Remark that even if the TOE shall provide access to the appropriate TSFs, it is still the responsibility of the applets to use them. Keys, PIN's are particular cases of an application's sensitive data (the Java Card System may possess keys as well) that ask for appropriate management (O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION). If the PIN class of the Java Card API is used, the objective (O.FIREWALL) shall contribute in covering this threat by controlling the sharing of the global PIN between the applets.

Other application data that is sent to the applet as clear text arrives to the APDU buffer, which is a resource shared by all applications. The disclosure of such data is prevented by the security objective O.GLOBAL_ARRAYS_CONFID.

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

T.CONFID-JCS-CODE 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 proof (OE.CODE-Evidence) of the check; 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 and the OE.CODE-EVIDENCE 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 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 and by the OE.CODE-EVIDENCE.

The objectives O.CARD-MANAGEMENT and OE.VERIFICATION and by the OE.CODE-EVIDENCE contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively.

T.INTEG-APPLI-CODE.LOAD This threat is countered by the security objective O.CARD-MANAGEMENT which ensures that the loading of packages is done securely and thus preserves the integrity of packages code.

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.

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.

T.INTEG-APPLI-DATA This threat is countered by bytecode verification and its proof (OE.VERIFICATION and OE.CODE-EVIDENCE) 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, OE.VERIFICATION and OE.CODE-EVIDENCE 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.

Concerning the confidentiality and integrity of application sensitive data, as applets may need to share some data or communicate with the CAD, cryptographic functions are required to actually protect the exchanged information (O.CIPHER). Remark that even if the TOE shall provide access to the appropriate TSFs, it is still the responsibility of the applets to use them. Keys and PIN's are particular cases of an application's sensitive data (the Java Card System may possess keys as well) that ask for appropriate management (O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION).

If the PIN class of the Java Card API is used, the objective (O.FIREWALL) is also concerned.

Other application data that is sent to the applet as clear text arrives to the APDU buffer, which is a resource shared by all applications. The integrity of the information stored in that buffer is ensured by the objective O.GLOBAL_ARRAYS_INTEG.

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

T.INTEG-APPLI-DATA.LOAD This threat is countered by the security objective O.SCP.SUPPORT 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.

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.

T.INTEG-JCS-CODE 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 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 2 objectives for the environment OE.VERIFICATION and OE.CODE-EVIDENCE.

The objectives O.CARD-MANAGEMENT, OE.VERIFICATION and OE.CODE-EVIDENCE contribute to cover this threat by controlling the access to card management functions and by checking the bytecode, respectively.

T.INTEG-JCS-DATA This threat is countered by bytecode verification (OE.VERIFICATION), the proof of integrity and authenticity (OE.CODE-EVIDENCE) 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, OE.VERIFICATION and OE.CODE-EVIDENCE 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.

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.

The objective OE.CODE-Evidence ensures that the application code cannot be modified between verification and installation. The evidence existence demonstrates that once the application code has been verified it cannot be changed after this verification.

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

OE.CODE-Evidence ensures that loaded applications are authorised applications to be loaded (The evidence proves that there is no modification between the verification and the loading).

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 objectives OE.VERIFICATION. This threat particularly concerns the security aspect of 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 2 objectives OE.VERIFICATION. This threat particularly concerns those points of the security aspect related to control flow confinement and the validity of the method references used in the bytecodes.

T.NATIVE This threat is countered by O.NATIVE which ensures that a Java Card applet can only access native methods indirectly that is, through an API. OE.APPLET also covers this threat by ensuring that no native applets shall be loaded in post-issuance. In addition to this, the bytecode verifier also prevents the program counter of an applet to jump into a piece of native code by confining the control flow to the currently executed method (OE.VERIFICATION).

T.EXE-CODE-REMOTE The O.REMOTE security objective contributes to prevent the invocation of a method that is not supposed to be accessible from outside the card.

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. Installation of malicious applications is blocked by the OE.CODE-EVIDENCE, this objective ensures with the existing evidence, that integrity of the applications is maintained after the verification of the application and before its installation.

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 Protection Profile, 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.

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.

CARD MANAGEMENT

T.INSTALL This threat is covered by the O.INSTALL security objective which ensures that installation is secure.

The objective O.CARD-MANAGEMENT controls the access to card management functions and thus contributes to cover this threat. 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 OE.CODE-Evidence covers T.INSTALL as it ensures that only checked application can be installed in the TOE. As the evidence shows that the is no application byte code modification between its verification and its loading and then its installation.

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 and OE.CODE-Evidence (by ensuring that loaded applets are authorized) control the access to card management functions and thus contributes to cover this threat.

5.3.1.3 DESFire

T.DF_DATA_MODIFICATION

The attacker deletes an applet or a package already in use on the card, or uses the deletion function to pave the way for further attacks (putting the TOE in an insecure state).

According to threat T.DF_Data-Modification the TOE shall avoid that user data stored by the TOE may be modified by unauthorised subjects. The objective O.DF_Access-Control requires an access control mechanism that limits the ability to modify data elements stored by the TOE. O.DF_Type_Consistency ensures that data types are adhered, so that data cannot be modified by abusing type-specific operations.

The terminal must support this by checking the TOE responses, which is required by OE.DF_Terminal_Support.

T.DF_IMPERSONATE

Impersonating authorized users during the authentication process of the MIFARE DESFire.

An unauthorized subject may try to impersonate an authorized subject during the authentication sequence of the MIFARE DESFire, e.g. by a man-in-the middle or replay attack.

The goal of O.DF_Authentication is that an authentication mechanism is implemented in the TOE that prevents these attacks.

Therefore this Threat is averted by the security objective O.DF_AUTHENTICATION. This must be supported by OE.DF_SECURE_VALUES because the authentication is based on keys and knowledge of the keys must be limited to the authorized users.

T.DF_CLONING

Cloning using keys, files and values maintained MIFARE DESFire.

Keys, files and values maintained by the MIFARE DESFire stored on the TOE may be read out by an unauthorized subject in order to create a duplicate.

This Threat is averted by the security objective O.DF_Access-Control that prevents the disclosure of sensitive data of the TOE and the security objective O.DF_AUTHENTICATION that limits the access to authorized user only. An appropriate key management according to OE.DF_SECURE_VALUES must be ensured.

5.3.2 Organizational Security Policies

5.3.2.1 (U)SIM part

Basic and Secure Applications Policies

OSP.SECURE-APPS-CERTIFICATION This OSP is enforced by the security objective for the operational environment of the TOE OE.SECURE-APPS-CERTIFICATION.

OSP.BASIC-APPS-VALIDATION This OSP is enforced by the security objective for the operational environment of the TOE OE.BASIC-APPS-VALIDATION.

OSP.SHARE-CONTROL This OSP is directly enforced by the security objective for the operational environment of the TOE OE.SHARE-CONTROL.

OSP.AID-MANAGEMENT This OSP is directly enforced by the security objective for the operational environment of the TOE OE.AID-MANAGEMENT.

Loading Policies

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

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

Key Policies

OSP.APSD-KEYS This OSP is enforced by the security objective for the operational environment of the TOE OE.AP-KEYS.

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

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

OSP.CASD-KEYS This OSP is enforced by the security objective for the operational environment of the TOE OE.CA-KEYS.

OSP.VASD-KEYS This OSP is enforced by the security objective for the operational environment of the TOE OE.VA-KEYS.

Platform

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

GlobalPlatform

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

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

5.3.2.2 Java Card System Protection Profile - Open Configuration

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.

5.3.2.3 DESFIRE part

The policy **OSP.DF_Confidentiality** requires the TOE to provide the possibility to protect selected data elements from eavesdropping during contact-less communication. In addition the data transfer is protected in a way that injected and bogus commands within the communication session before the protected data transfer can be detected. The terminal must support this by checking the TOE responses, which is required by OE.DF_Terminal_Support. Since O.DF_Confidentiality requires that the security attribute for a data element contains an option that the communication related to this data element must be encrypted and protected and because OE.DF_Terminal_Support ensures the support by the terminal, the two objectives cover the policy.

According to the policy **OSP.DF_Transaction** the TOE shall be able to provide the possibility to combine a number of data modification operations in one transaction, so that either all operations or no operation at all is performed. This is exactly the goal of the objective **O.DF_Transaction**, therefore theis policy is covered by **O.DF_Transaction**.

The objectives **O.DF_Authentication** and **O.DF_Access-Control** provide means to authorise subjects and to implement access control to data elements in a way that unauthorised subjects cannot read any element usable for tracing. Therefore the policy is covered by the three objectives.

5.3.3 Assumptions

5.3.3.1 Actors

A.MOBILE-OPERATOR This assumption is directly upheld by OE.MOBILE-OPERATOR.

A.OTA-ADMIN This assumption is directly upheld by OE.OTA-ADMIN.

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

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

This assumption is also upheld by the security objective of the environment OE.CODE-EVIDENCE which ensures that evidences exist that the application code has been verified and not changed after verification.

A.KEY-ESCROW This assumption is directly upheld by OE.KEY-ESCROW.

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

5.3.3.2 Java Card System Protection Profile - Open Configuration

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

A.VERIFICATION This assumption is upheld by the security objective on the operational environment OE.VERIFICATION which guarantees that all the bytecodes shall be verified at least once, before the loading, before the installation or before the execution in order to ensure that each bytecode is valid at execution time.

This assumption is also upheld by the security objective of the environment OE.CODE-EVIDENCE which ensures that evidences exist that the application code has been verified and not changed after verification.

5.3.3.3 DESFire

A.DF_Secure _Value is assumption is directly upheld by OE.DF_Secure_Values

A.DF_Terminal_Support is assumption is directly upheld by OE.DF_Terminal_Support.

5.3.4 SPD and Security Objectives

Threats	Security Objectives	Rationale in section
T.PHYSICAL	O.SCP.IC, O.SCP.SUPPORT	5.3.1
T.INTEG-USER-DATA	O.DOMAIN-RIGHTS, OE.CA-KEYS, OE.AP-KEYS, OE.VA_KEYS	5.3.1
T.COM_EXPLOIT	O.COMM_AUTH, O.COMM_INTEGRITY, O.COMM_CONFIDENTIALITY	5.3.1
T.UNAUTHORIZED_CARD_MNGT	O.CARD-MANAGEMENT, O.COMM_AUTH, O.COMM_INTEGRITY, O.APPLI-AUTH, O.DOMAIN-RIGHTS	5.3.1
T.LIFE_CYCLE	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS	5.3.1
T.UNAUTHORIZED_ACCESS	OE.SHARE-CONTROL	5.3.1
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.SUPPORT, O.SCP.RECOVERY, O.CARD-MANAGEMENT	5.3.1
T.CONFID-JCS-CODE	OE.VERIFICATION, O.NATIVE, O.CARD-MANAGEMENT	5.3.1
T.CONFID-JCS-DATA	OE.VERIFICATION, O.SID, O.OPERATE, O.FIREWALL, O.ALARM, O.SCP.SUPPORT, O.SCP.RECOVERY, O.CARD-MANAGEMENT	5.3.1
T.INTEG-APPLI-CODE	OE.VERIFICATION, O.NATIVE, O.CARD-MANAGEMENT,OE.CODE-EVIDENCE	5.3.1
T.INTEG-APPLI-CODE.LOAD	O.CARD-MANAGEMENT, O.LOAD, OE.CODE-EVIDENCE	5.3.1
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.SUPPORT, O.SCP.RECOVERY, O.CARD-MANAGEMENT, OE.CODE-EVIDENCE	5.3.1
T.INTEG-APPLI-DATA.LOAD	O.SCP.SUPPORT, O.CARD-MANAGEMENT, O.LOAD, OE.CODE-EVIDENCE	5.3.1
T.INTEG-JCS-CODE	OE.VERIFICATION, O.NATIVE, O.CARD-MANAGEMENT, OE.CODE-EVIDENCE	5.3.1
T.INTEG-JCS-DATA	OE.VERIFICATION, O.SID, O.OPERATE, O.FIREWALL, O.ALARM, O.SCP.SUPPORT, O.SCP.RECOVERY, O.CARD-MANAGEMENT, OE.CODE-EVIDENCE	5.3.1

Threats	Security Objectives	Rationale in section
T.SID.1	O.FIREWALL, O.GLOBAL_ARRAYS_CONFID, O.GLOBAL_ARRAYS_INTEG, O.SID, O.INSTALL, O.CARD-MANAGEMENT	5.3.1
T.SID.2	O.SID, O.OPERATE, O.FIREWALL, O.SCP.SUPPORT, O.SCP.RECOVERY, O.INSTALL	5.3.1
T.EXE-CODE.1	OE.VERIFICATION, O.FIREWALL	5.3.1
T.EXE-CODE.2	OE.VERIFICATION	5.3.1
T.NATIVE	OE.VERIFICATION, OE.APPLET, O.NATIVE	5.3.1
T.EXE-CODE-REMOTE	O.REMOTE	5.3.1
T.RESOURCES	O.OPERATE, O.RESOURCES, O.SCP.SUPPORT, O.SCP.RECOVERY, O.INSTALL	5.3.1
T.OBJ-DELETION	O.OBJ-DELETION	5.3.1
T.INSTALL	O.CARD-MANAGEMENT, O.INSTALL, O.LOAD	5.3.1
T.DELETION	O.CARD-MANAGEMENT, O.DELETION	5.3.1
T.DF_DATA_MODIFICATION	O.DF_ACCESS-CONTROL; O.DF_TYPE_CONSISTENCY	5.3.1
T.DF_IMPERSONATE	O.DF_AUTHENTICATION	5.3.1
T.DF_CLONING	O.DF_ACCESS-CONTROL; O.DF_AUTHENTICATION	5.3.1

Table 9: Threats and Security Objectives - Coverage

Security Objectives	Threats
O.CARD-MANAGEMENT	T.UNAUTHORIZED_CARD_MNGT, T.LIFE_CYCLE, 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.INSTALL, T.DELETION
O.DOMAIN-RIGHTS	T.INTEG-USER-DATA, T.UNAUTHORIZED_CARD_MNGT, T.LIFE_CYCLE
O.APPLI-AUTH	T.UNAUTHORIZED_CARD_MNGT
O.COMM_AUTH	T.COM_EXPLOIT, T.UNAUTHORIZED_CARD_MNGT
O.COMM_INTEGRITY	T.COM_EXPLOIT, T.UNAUTHORIZED_CARD_MNGT
O.COMM_CONFIDENTIALITY	T.COM_EXPLOIT
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.OPERATE	T.CONFID-APPLI-DATA, T.CONFID-JCS-DATA, T.INTEG-APPLI-DATA, T.INTEG-JCS-DATA, T.SID.2, T.RESOURCES
O.REALLOCATION	T.CONFID-APPLI-DATA, T.INTEG-APPLI-DATA
O.RESOURCES	T.RESOURCES
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.REMOTE	T.EXE-CODE-REMOTE
O.OBJ-DELETION	T.OBJ-DELETION
O.DELETION	T.DELETION

Security Objectives	Threats
O.LOAD	T.INTEG-APPLI-CODE.LOAD, T.INTEG-APPLI-DATA.LOAD, T.INSTALL
O.INSTALL	T.SID.1, T.SID.2, T.RESOURCES, T.INSTALL
O.SCP.SUPPORT	T.PHYSICAL, T.CONFID-APPLI-DATA, T.CONFID-JCS-DATA, T.INTEG-APPLI-DATA, T.INTEG-APPLI-DATA.LOAD, T.INTEG-JCS-DATA, T.SID.2, T.RESOURCES T.INTEG-USER-DATA
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.DF_Access-Control	T.DF_Data-Modification T.DF_Cloning
O. DF_Type-Consistency	T.DF_Data-Modification
O.DF_Confidentiality	OSP.DF_Confidentiality
O.DF_Authentication	T.DF_Impersonate, T.DF_Cloning
O.DF_Transaction	OSP.DF_Transaction
OE.DF_Terminal-Support	T.DF_Data-Modification OSP.DF_Confidentiality
OE.MOBILE-OPERATOR	
OE.OTA-ADMIN	
OE.APPS-PROVIDER	
OE.VERIFICATION-AUTHORITY	
OE.KEY-ESCROW	
OE.CONTROLLING-AUTHORITY	
OE.SECURE-APPS-CERTIFICATION	
OE.BASIC-APPS-VALIDATION	
OE.AID-MANAGEMENT	
OE.OTA-LOADING	
OE.OTA-SERVERS	
OE.AP-KEYS	T.INTEG-USER-DATA
OE.OPERATOR-KEYS	
OE.KEY-GENERATION	

Security Objectives	Threats
OE.CA-KEYS	T.INTEG-USER-DATA
OE.VA-KEYS	T.INTEG-USER-DATA
OE.KEY-CHANGE	
OE.SECURITY-DOMAINS	
OE.QUOTAS	
OE.SHARE-CONTROL	T.UNAUTHORIZED_ACCESS
OE.APPLET	T.NATIVE
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
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, T.SID.1, T.SID.2,T.RESOURCES,T.DELETION,T.INSTALL

Table 10: Security Objectives and Threats - Coverage

Organisational Security Policies	Security Objectives	Rationale In Section
OSP.SECURE-APPS-CERTIFICATION	OE.SECURE-APPS-CERTIFICATION	5.3.2
OSP.BASIC-APPS-VALIDATION	OE.BASIC-APPS-VALIDATION	5.3.2
OSP.SHARE-CONTROL	OE.SHARE-CONTROL	5.3.2
OSP.AID-MANAGEMENT	OE.AID-MANAGEMENT	5.3.2
OSP.OTA-LOADING	OE.OTA-LOADING	5.3.2
OSP.OTA-SERVERS	OE.OTA-SERVERS	5.3.2
OSP.APSD-KEYS	OE.AP-KEYS	5.3.2
OSP.OPERATOR-KEYS	OE.OPERATOR-KEYS	5.3.2
OSP.KEY-GENERATION	OE.KEY-GENERATION	5.3.2
OSP.CASD-KEYS	OE.CA-KEYS	5.3.2
OSP.VASD-KEYS	OE.VA-KEYS	5.3.2
OSP.KEY-CHANGE	OE.KEY-CHANGE	5.3.2
OSP.SECURITY-DOMAINS	OE.SECURITY-DOMAINS	5.3.2
OSP.QUOTAS	OE.QUOTAS	5.3.2
OSP.VERIFICATION	OE.VERIFICATION, O.LOAD, OE.CODE-EVIDENCE	5.3.2

Table 11: OSPs and Security Objectives - Coverage

Security Objectives	Organisational Security Policies
O.CARD-MANAGEMENT	
O.DOMAIN-RIGHTS	
O.APPLI-AUTH	
O.COMM_AUTH	
O.COMM_INTEGRITY	
O.COMM_CONFIDENTIALITY	
O.SID	
O.FIREWALL	
O.GLOBAL_ARRAYS_CONFID	
O.GLOBAL_ARRAYS_INTEG	
O.NATIVE	
O.OPERATE	
O.REALLOCATION	
O.RESOURCES	
O.ALARM	
O.CIPHER	
O.KEY-MNGT	
O.PIN-MNGT	
O.TRANSACTION	
O.REMOTE	
O.OBJ-DELETION	
O.DELETION	
O.LOAD	
O.INSTALL	
O.SCP.SUPPORT	
O.SCP.IC	
O.SCP.RECOVERY	
OE.MOBILE-OPERATOR	
OE.OTA-ADMIN	
OE.APPS-PROVIDER	
OE.VERIFICATION-AUTHORITY	
OE.KEY-ESCROW	
OE.CONTROLLING-AUTHORITY	
OE.SECURE-APPS-CERTIFICATION	OSP.SECURE-APPS-CERTIFICATION

Security Objectives	Organisational Security Policies
OE.BASIC-APPS-VALIDATION	OSP.BASIC-APPS-VALIDATION
OE.AID-MANAGEMENT	OSP.AID-MANAGEMENT
OE.OTA-LOADING	OSP.OTA-LOADING
OE.OTA-SERVERS	OSP.OTA-SERVERS
OE.AP-KEYS	OSP.APSD-KEYS
OE.OPERATOR-KEYS	OSP.OPERATOR-KEYS
OE.KEY-GENERATION	OSP.KEY-GENERATION
OE.CA-KEYS	OSP.CASD-KEYS
OE.VA-KEYS	OSP.VASD-KEYS
OE.KEY-CHANGE	OSP.KEY-CHANGE
OE.SECURITY-DOMAINS	OSP.SECURITY-DOMAINS
OE.QUOTAS	OSP.QUOTAS
OE.SHARE-CONTROL	OSP.SHARE-CONTROL
OE.APPLET	
OE.VERIFICATION	OSP.VERIFICATION
OE.CODE-EVIDENCE	OSP.VERIFICATION

Table 12: Security Objectives and OSPs – Coverage

Assumptions	Security objectives for the Operational Environment	Rationale In Section
A.MOBILE-OPERATOR	OE.MOBILE-OPERATOR	5.3.3
A.OTA-ADMIN	OE.OTA-ADMIN	5.3.3
A.APPS-PROVIDER	OE.APPS-PROVIDER	5.3.3
A.VERIFICATION-AUTHORITY	OE.VERIFICATION-AUTHORITY	5.3.3
A.KEY-ESCROW	OE.KEY-ESCROW	5.3.3
A.CONTROLLING-AUTHORITY	OE.CONTROLLING-AUTHORITY	5.3.3
A.APPLET	OE.APPLET	5.3.3
A.VERIFICATION	OE.VERIFICATION OE.CODE-EVIDENCE	5.3.3
A.DF_SECURE_VALUES	OE.DF_SECURE_VALUES	5.3.3
A.DF_TERMINAL_Support	OE.DF_Terminal_Support	5.3.3

Table 13: Assumptions and Security Objectives for the Operational Environment – Coverage

Security objectives for the Operational Environment	Assumptions
OE.MOBILE-OPERATOR	A.MOBILE-OPERATOR
OE.OTA-ADMIN	A.OTA-ADMIN
OE.APPS-PROVIDER	A.APPS-PROVIDER
OE.VERIFICATION-AUTHORITY	A.VERIFICATION-AUTHORITY
OE.KEY-ESCROW	A.KEY-ESCROW
OE.CONTROLLING-AUTHORITY	A.CONTROLLING-AUTHORITY
OE.SECURE-APPS-CERTIFICATION	
OE.BASIC-APPS-VALIDATION	
OE.AID-MANAGEMENT	
OE.OTA-LOADING	
OE.OTA-SERVERS	
OE.AP-KEYS	
OE.OPERATOR-KEYS	
OE.KEY-GENERATION	
OE.CA-KEYS	
OE.VA-KEYS	
OE.KEY-CHANGE	
OE.SECURITY-DOMAINS	
OE.QUOTAS	
OE.SHARE-CONTROL	
OE.APPLET	A.APPLET
OE.VERIFICATION	A.VERIFICATION
OE.CODE-EVIDENCE	A.VERIFICATION
OE.DF_Secure_Values	A.DF_Secure_Value
OE.DF_Terminal_Support	A.DF_Terminal_Support

Table 14: Security Objectives for the Operational Environment and Assumptions - Coverage

6 Extended requirements

6.1 Extended families

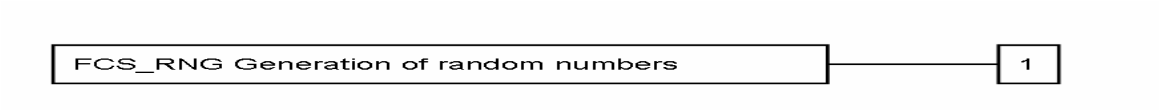
6.1.1 Extended family FCS_RNG - Generation of random numbers

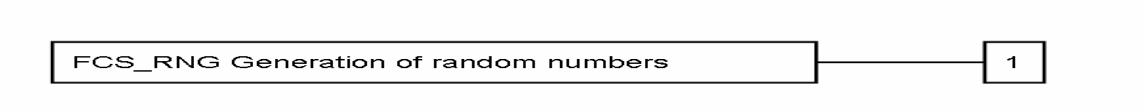
6.1.1.1 Description

The description is issued from FCS_ see [28] section 5.1.

Family behaviour

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





FCS_RNG.1 Generation of random numbers requires that random numbers meet a defined quality metric.

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

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

FCS_RNG.1 Random number generation

Hierarchical to: No other components.

Dependencies: No dependencies.

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

FCS_RNG.1.2 The TSF shall provide random numbers that meet [assignment: a defined quality metric].

Application Note 11: 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 pseudo random output. A hybrid RNG combines the principles of physical and deterministic RNGs.

7 Security Functional Requirements

7.1 Security Functional Requirements

This section describes the requirements imposed on the TOE in order to achieve the security objectives laid down in the previous chapter. Except FCS_RNG, extended requirement, all the requirements identified in this section are instances of those stated in [2].

This chapter is divided in 5 parts:

- the first is related to (U)SIM Platform,
- the second is related to the Java Card System Platform security functional requirements,
- the third is related to DESFire requirements,
- the forth is related to IC requirements,
- and the fifth expresses SCPG requirements.

7.1.1 (U)SIM part of the TOE

This section describes the SFR for (U)SIM part of the TOE.

7.1.1.1 Card Manager (CMGRG)

This section contains the security requirements for the card manager.

The security requirements below help to define a policy for controlling access to card content management operations and for expressing card issuer security concerns. Most of them come from [JCS] but are instantiated to add more precisions regarding (U)SIM card content management. This policy depends on the particular security and card management architecture present in the card. Therefore the policy shall be instantiated when developing conformant Security Targets.

Card Content Management

FDP_UIT.1/CCM Data exchange integrity

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

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

FDP_ROL.1/CCM Basic rollback

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

FDP_ROL.1.2/CCM The TSF shall permit operations to be rolled back within the **scope of install() within the bounds of the Commit Capacity ([7], §7.8), and those described in [6]**.

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 [6] (see for instance, PIN-blocking, PIN-checking, update of Transient objects).

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

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

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

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

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

FDP_ITC.2.5/CCM The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: **The TSF shall use the PUT KEY data format or the STORE DATA data format when interpreting the user data from outside the TOE**.

Application note:

This Functional Component Instance enforces a security information flow control policy. Rules must be defined for importation operations. These rules must take into account all user data.

FPT_FLS.1/CCM Failure with preservation of secure state

FPT_FLS.1.1/CCM The TSF shall preserve a secure st-1e when the following types of failures occur: **the Security Domain fails to load/install/delete an Executable File / application instance as described in [7], Section 11.3.4**.

FCS_COP.1/DAP Cryptographic operation

FCS_COP.1.1/DAP_1 The TSF shall perform **verification of the DAP signature attached to Executable Load Applications** in accordance with a specified cryptographic algorithm

- **PKC Scheme: SHA-1 hash[26] and PKCS#1 RSA signature[26]**

and cryptographic key sizes

- **PKC Scheme: RSA key length 1024 bits**

that meet the following:

- **Sections C.1.2 and C.6 of** [9]
- **PKC Scheme: SSA-PKCS1 as defined in PKCS#1-v2_1 [26]**

FCS_COP.1.1/DAP_2 The TSF shall perform **verification of the DAP signature attached to Executable Load Applications** in accordance with a specified cryptographic algorithm

- **PKC Scheme: SHA-1 hash[26] and PKCS#1 RSA signature[26]**

and cryptographic key sizes

- **PKC Scheme: RSA key length 2048 bits**

that meet the following:

- **Sections C.1.2 and C.6 of** [9]
- **PKC Scheme: SSA-PSS as defined in PKCS#1-v2_1 [26]**

FCS_COP.1/TOKEN-OT_TDES Cryptographic operation

FCS_COP.1.1/TOKEN-OT _TDES The TSF shall perform **verification of the Token signature attached to the card content management commands** in accordance with a specified cryptographic algorithm

- **Single DES plus final Triple DES MAC (ISO/IEC 9797-1)**

and cryptographic key sizes

- **TDES 2 keys**

that meet the following:

- **Sections B.1.2.2 and B.4 of** [9]

FCS_COP.1/RECEIPTS-OT_TDES Cryptographic operation

FCS_COP.1.1/RECEIPTS-OT_TDES The TSF shall perform **computation of the Receipt signature sent back in the response associated to the card content management commands** in accordance with a specified cryptographic algorithm

- **DES Scheme: Single DES plus final Triple DES MAC (Retail MAC)**

and cryptographic key sizes

- **TDES 2 keys**

that meet the following:

- **Sections C.1.1.2 and C.5 of** [9]
- **DES Scheme: ISO 9797-1 as MAC Algorithm 3 with output transformation 3, without truncation, and with DES taking the place of the block cipher**

FCS_COP.1/TOKEN-OT_AES Cryptographic operation

FCS_COP.1.1/TOKEN-OT _AES The TSF shall perform **verification of the Token signature attached to the card content management commands** in accordance with a specified cryptographic algorithm

- **AES (NIST 800-38B)**

and cryptographic key sizes

- **AES key length 128 bits keys**

that meet the following:

- **Sections 4.4** [9]

FCS_COP.1/RECEIPTS-OT_AES Cryptographic operation

FCS_COP.1.1/RECEIPTS-OT_AES The TSF shall perform **computation of the Receipt signature sent back in the response associated to the card content management commands** in accordance with a specified cryptographic algorithm

- **AES** algorithm **(NIST 800-38B)**

and cryptographic key sizes

- **128 bits keys**

that meet the following:

- **Sections 4.8** [9]

FCS_COP.1/TOKEN-OT_RSA Cryptographic operation

FCS_COP.1.1/TOKEN-OT _RSA The TSF shall perform **verification of the Token signature attached to the card content management commands** in accordance with a specified cryptographic algorithm

- **PKC Scheme: SHA-1 hash and PKCS#1 RSA signature[26]**

and cryptographic key sizes

- **PKC Scheme: RSA key length 1024 bits**

that meet the following:

- **Sections C.1.1.1 and C.4 of** [9]
- **PKC Scheme: RSA-PKCS1-v1_5 as defined in PKCS#1-v2_1**

FCS_COP.1/CIPHERLOADFILE-OT_TDES Cryptographic operation

FCS_COP.1.1/CIPHERLOADFILE-OT The TSF shall perform **decryption of the Ciphered Load File Data Block** in accordance with a specified cryptographic algorithm

- **DES Scheme: Triple DES 2 Keys CBC (with IV = 0 and padding '80…00')**

and cryptographic key sizes

- **TDES 2 keys**

that meet the following:

- **Sections C.1.1.2 and C.5 of [9]**
- **DES Scheme: ISO 9797- padding M2**

FCS_COP.1/CIPHERLOADFILE-OT_AES Cryptographic operation

FCS_COP.1.1/CIPHERLOADFILE-OT The TSF shall perform **decryption of the Ciphered Load File Data Block** in accordance with a specified cryptographic algorithm

- **AES Scheme**

and cryptographic key sizes

- **keys 128**

that meet the following:

FIPS 197 [21]

Application note: for the decryption, the 2 algorithms are available in the TOE, an SD can require a TDES and another SD an AES. It depends of key loaded in the personalisation of the security Domain.

FPT_TDC.1/CCM-OT Inter-TSF basic TSF data consistency

FPT_TDC.1.1/CCM-OT The TSF shall provide the capability to consistently interpret **Keys attributes specification** when shared between the TSF and another trusted IT product.

FPT_TDC.1.2/CCM-OT The TSF shall use **PUT KEY and the STORE DATA specification [9]** when interpreting the TSF data from another trusted IT product.

Security Domain

FDP_ACF.1/SD Security attributes based access control

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

- **Subjects:**
  - **S.INSTALLER, defined in [5] and represented by the GlobalPlatform Environment (OPEN) on the card, the Card Life Cycle attributes (defined in Section 5.1.1 of [9]);**
  - **S.ADEL, also defined in [5] and represented by the GlobalPlatform Environment (OPEN) on the card;**
  - **S.SD receiving the Card Content Management commands (through APDUs or APIs) with a set of privileges (defined in Section 6.6.1 of [9]), a life cycle status (defined in Section 5.3.2 of [9]) and a Secure Communication Security level (defined in Section 10.6 of [9]);**
  - **S.CAD, defined in [5], the off-card entity that communicates with the S.INSTALLER through S.SD;**
- **Objects:**
  - **The Delegation Token, in case of Delegated Management operations, with the attributes Present or Not Present;**
  - **The DAP Block, in case of application loading, with the attributes Present or Not Present;**
  - **The Executable Load File, Executable Module or applet instance, in case of application loading, installation, extradition or registry update, with a set of intended privileges and its targeted associated SD AID**.

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

Runtime behaviour rules defined by GlobalPlatform for:

- **loading (Section 9.3.5 of [9]);**
- **installation (Section 9.3.6 of [9]);**
- **extradition (Section 9.4.1 of [9]);**
- **registry update (Section 9.4.2 of [9]);**
- **content removal (Section 9.5 of [9])**.

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

FDP_ACF.1.4/SD The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **following rule: when at least one of the rules defined by GlobalPlatform does not hold**.

FMT_SMR.1/SD Security roles

FMT_SMR.1.1/SD The TSF shall maintain the roles**: -the Card Issuer, -the Application Provider, -the Controlling Authority**.

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

Application note:

Security Domain: On-card entity providing support for the control, security, and communication requirements of an off-card entity (the Card Issuer, an Application Provider or a Controlling Authority).

FMT_MSA.3/SD Static attribute initialization

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

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

Refinement:

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

Application note:

When the Security Domain life cycle state is LOCKED, card content management is forbidden. An application instance can not unlock itself.

FMT_MSA.1/SD Management of security attributes

FMT_MSA.1.1/SD The TSF shall enforce the **Security Domain access control policy** to restrict the ability to **query and modify** the security attributes **AS.CMLIFECYC**

"Operation -- Security attributes -- Role"

"Query -- AS.CMLIFECYC -- Security Domain Application Instance"

"Modify -- AS.CMLIFECYC -- Security Domain Application Instance"

to **the Security Domain and the application instance itself**.

Application note:

AS.CMLIFECYC: security attribute representing the card life Cycle state.

FMT_SMF.1/SD Specification of Management Functions

FMT_SMF.1.1/SD The TSF shall be capable of performing the following management functions: **Query and Modify the Security Domain access control policy**.

FDP_ACC.1/SD Subset access control

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

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

Secure Channel

FTP_ITC.1/SC Inter-TSF trusted channel

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

FTP_ITC.1.2/SC The TSF shall permit **another trusted IT product** to initiate communication via the trusted channel.

FTP_ITC.1.3/SC The TSF shall initiate communication via the trusted channel for **all card management functions:**

- **loading;**
- **installation;**
- **extradition;**
- **registry update;**
- **deletion;**
- **SD personalization**

FCO_NRO.2/SC Enforced proof of origin

FCO_NRO.2.1/SC The TSF shall enforce the generation of evidence of origin for transmitted **Executable load files** at all times.

FCO_NRO.2.2/SC The TSF shall be able to relate the **DAP block** of the originator of the information, and the **identity** of the information to which the evidence applies.

FCO_NRO.2.3/SC The TSF shall provide a capability to verify the evidence of origin of information to **originator** given **Executable load files**.

FDP_IFC.2/SC Complete information flow control

FDP_IFC.2.1/SC The TSF shall enforce the **Secure Channel Protocol information flow control policy** on

- **the subjects S.CAD and S.SD, involved in the exchange of messages between the (U)SIM card and the CAD through a potentially unsafe communication channel**
- **the information controlled by this policy is the card content management command, including personalization commands, in the APDUs sent to the card and their associated responses returned to the CAD**

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

FDP_IFC.2.2/SC The TSF shall ensure that all operations that cause any information in the TOE to flow to and from any subject in the TOE are covered by an information flow control SFP.

Application note:

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

Operation Description OP.SEND(M) A subject sends a message M through the communication channel. OP.RECEIVE(M) A subject receives a message M from the communication channel.

The information (prefixed with an "I") controlled by the typing policy is the APDUs exchanged by the subjects through the communication channel linking the (U)SIM card and the CAD.

Information Description I.APDU Any APDU sent to or from the card through the communication channel.

FDP_IFF.1/SC Simple security attributes

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

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

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

- **Runtime behaviour rules defined by GlobalPlatform for:**
  - **loading (Section 9.3.5 of [9]);**
  - **installation (Section 9.3.6 of [9]);**
  - **extradition (Section 9.4.1 of [9]);**
  - **registry update (Section 9.4.2 of [9]);**
  - **SD personalization rules, Pull and Push model (Section 11 of [12])**.

FDP_IFF.1.3/SC The TSF shall enforce the following rules: **none**.

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

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

- **When none of the conditions listed in the element FDP_IFF.1.4 of this component hold and at least one of those listed in the element FDP_IFF.1.2 does not hold**.

Application note:

The on-card and the off-card subjects have security attributes such as MAC, Cryptogram, Challenge, Key Set, Static Keys, etc.

FMT_MSA.3/SC Static attribute initialization

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

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

Refinement:

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

FMT_SMF.1/SC Specification of Management Functions

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

- **Management functions specified in GlobalPlatform specifications [9]:**
  - **loading (Section 9.3.5 of [9]);**
  - **installation (Section 9.3.6 of [9]);**
  - **extradition (Section 9.4.1 of [9]);**
  - **registry update (Section 9.4.2 of [9]);**
  - **deletion (Section 9.5 of [9]);**
  - **SD personalization rules, Pull and Push model (Section 11 of [12]).**

Application note:

All management functions related to SCP02 secure channel shall be relevant.

FIA_UID.1/SC Timing of identification

FIA_UID.1.1/SC The TSF shall allow

- **application selection;**
- **initializing a secure channel with the card;**
- **requesting data that identifies the card or the Card Issuer;**

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

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

Application note:

The GlobalPlatform TSF mediated actions listed in [9] such as selecting an application, requesting data, initializing, etc.

FIA_UAU.1/SC Timing of authentication

FIA_UAU.1.1/SC The TSF shall allow **the TSF mediated actions listed in FIA_UID.1/SC** on behalf of the user to be performed before the user is authenticated.

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

FIA_UAU.4/SC Single-use authentication mechanisms

FIA_UAU.4.1/SC The TSF shall prevent reuse of authentication data related to **the authentication mechanism used to open a secure communication channel with the card**.

FIA_AFL.1/SC-OT Authentication failure handling

FIA_AFL.1.1/SC-OT The TSF shall detect when **1** unsuccessful authentication attempts occur related to **the opening of a secure communication channel with the card**.

FIA_AFL.1.2/SC-OT When the defined number of unsuccessful authentication attempts has been **surpassed**, the TSF shall **slow down the next authentication. The waiting time is augmented with a maximum number of unsuccessful authentications of 15**.

FIA_UAU.7/SC-OT Protected authentication feedback

FIA_UAU.7.1/SC-OT The TSF shall provide only **the result of the authentication (NOK), the key set version, Secure Channel identifier and**, **the card random and the card cryptogram** to the user while the authentication is in progress.

FPR_UNO.1/SC-OT Unobservability

FPR_UNO.1.1/SC-OT The TSF shall ensure that **all subjects** are unable to observe the operation **import, use** on **AS.KEYSET_VALUE** by **Security Domain**.

Application note:

AS.KEY_VALUE: value of the Key imported or in use.

FMT_MSA.1/SC Management of security attributes

FMT_MSA.1.1/SC The TSF shall enforce the **Secure Channel Protocol (SCP) information flow control policy** to restrict the ability to **delete and modify** the security attributes **AS.KEYSET_VERSION and AS.KEYSET_VALUE** to **Security Domain**.

Application note:

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

AS.KEY_VERSION: reference of the key. AS.KEY_VALUE: value of the Key.

7.1.2 Java Card System Protection Profile - Open Configuration

This section states the security functional requirements for the Java Card System - Open configuration.

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. This group is the union of requirements from the Core (*CoreG*) and the Logical channels (*LCG*).
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.
Remote Method Invocation (RMI)	The RMIG contains the security requirements for the remote method invocation feature, which provides a new protocol of communication between the terminal and the applets.
Object deletion (*ODELG*)	The ODELG contains the security requirements for the object deletion capability. This provides a safe memory recovering mechanism.
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.

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

Subject	Description
S.ADEL	The applet deletion manager which also acts on behalf of the card issuer. It may be an applet (§11), but its role asks anyway for a specific treatment from the security viewpoint. This subject is unique and is involved in the ADEL security policy defined in §7.1.3.1.
S.APPLET	Any applet instance.
S.BCV	The bytecode verifier (BCV), which acts on behalf of the verification authority who is in charge of the bytecode verification of the packages. This subject is involved in the PACKAGE LOADING security policy defined in §7.1.7.
S.CAD	The CAD represents the actor that requests, by issuing commands to the card, for RMI services. 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	Operand stacks 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.

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.
O.REMOTE_MTHD	A method of a remote interface.
O.REMOTE_OBJ	A remote object is an instance of a class that implements one (or more) remote interfaces. A remote interface is one that extends, directly or indirectly, the interface java.rmi.Remote ([6]).
O.RMI_SERVICE	These are instances of the class javacard.framework.service.RMIService. They are the objects that actually process the RMI services.
O.ROR	A remote object reference. It provides information concerning: (i) the identification of a remote object and (ii) the Implementation class of the object or the interfaces implemented by the class of the object. This is the object's information to which the CAD can access.

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.
I.RORD	Remote object reference descriptors which provide information concerning: (i) the identification of the remote object and (ii) the implementation class of the object or the interfaces implemented by the class of the object. The descriptor is the only object's information to which the CAD can access.

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".
Currently Active Context	Package AID or "Java Card RE".
Dependent package AID	Allows the retrieval of the Package AID and Applet's version number ([8], §4.5.2).
ExportedInfo	Boolean (indicates whether the remote object is exportable or not).
Identifier	The Identifier of a remote object or method is a number that uniquely identifies the remote object or method, respectively.
LC Selection Status	Multiselectable, Non-multiselectable or "None".
LifeTime	CLEAR_ON_DESELECT or PERSISTENT (*).
Owner	The Owner of an object is either the applet instance that created the object or the package (library) where it has been defined (these latter objects can only be arrays that initialize static fields of the package). The owner of a remote object is the applet instance that created the object.
Package AID	The AID of each package indicated in the export file.
Registered Applets	The set of AID of the applet instances registered on the card.
Remote	An object is Remote if it is an instance of a class that directly or indirectly implements the interface java.rmi.Remote.
Resident Packages	The set of AIDs of the packages already loaded on the card.
Returned References	The set of remote object references that have been sent to the CAD during the applet selection session. This attribute is implementation dependent.
Selected Applet Context	Package AID or "None".
Sharing	Standards, SIO, Java Card RE entry point or global array.
Static References	Static fields of a package may contain references to objects. The Static References attribute records those references.

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

Operations (prefixed with "OP") are described in the following table. Each operation has 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.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.GET_ROR(O.APPLET,...)	Retrieves the initial remote object reference of a RMI based applet. This reference is the seed which the CAD client application needs to begin remote method invocations.
OP.INSTANCE_FIELD(O.JAVAOBJECT, field)	Read/Write a field of an instance of a class in the Java programming language.
OP.INVK_VIRTUAL(O.JAVAOBJECT, method, arg1,...)	Invoke a virtual method (either on a class instance or an array object).
OP.INVK_INTERFACE(O.JAVAOBJECT, method, arg1,...)	Invoke an interface method.
OP.INVOKE(O.RMI_SERVICE,...)	Requests a remote method invocation on the remote object.
OP.JAVA(...)	Any access in the sense of [7], §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.PUT(S1,S2,I)	Transfer a piece of information I from S1 to S2.
OP.RET_RORD(S.JCRE,S.CAD,I.RORD)	Send a remote object reference descriptor to the CAD.
OP.THROW(O.JAVAOBJECT)	Throwing of an object (athrow, see [7], §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. For instance, during the creation of an object, the JavaCardClass attribute's value is chosen by the creator.

7.1.2.1 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.

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 ([7], §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 ([7], §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 ([7], §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 ([7], §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 authorize access of subjects to objects based on the following additional rules:

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

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

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

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 ([8], §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 ([7], §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).

([7], 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.

([7], §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 ([8], §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. ([7], §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 the following rules: **none**.

FDP_IFF.1.4/JCVM The TSF shall explicitly authorize 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 ([7], §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 [8], §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 [7], §4 and [8], §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 [7], §4 and [8], §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.
- An O.JAVAOBJECT whose Sharing attribute value is a global array necessarily has "array of primitive type" as a JavaCardClass security attribute's value.
- 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 initialization

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

FMT_MSA.3.2/FIREWALL [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 ([7], §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 initialization

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

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

FMT_SMF.1 Specification of Management Functions

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

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

FMT_SMR.1 Security roles

FMT_SMR.1.1 The TSF shall maintain the roles**:**

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

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

Application Programming Interface

The following SFRs are related to the Java Card API.

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

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

FCS_CKM.1 Cryptographic key generation

FCS_CKM.1.1 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 for algorithm	Cryptographic key size	list of standard for key generation
ECC	160, 192, 224, 256, 320, 384, 512, and 521 bits	None
RSA	from 1024 to 2048 bits with a step of 32-bit	FIPS 186-3 [20]

FCS_CKM.2 Cryptographic key distribution

FCS_CKM.2.1 The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method **command setKey** that meets the following: **[6] specification**.

Application note:

- Command SetKEY that meets [6] specification.

FCS_CKM.3 Cryptographic key access

FCS_CKM.3.1 The TSF shall perform **the following types of cryptographic key access** in accordance with a specified cryptographic key access method **see refinements below** that meets the following: **Standard: 1. See related document [7], chapter 9.5. Standard: 2. See related document [9], chapter 4.3. Standard: 3. See related document [6], packages "javacardx.security" and "javacardx.crypto"**.

Application note:

- The keys can be accessed as specified in [6] Key class.

Refinement: Type of cryptographic key access Cryptographic key access methods (or commands) DES The following commands: PUT_KEY, EXTERNAL AUTHENTICATE, INITIALIZE UPDATE,

The following "ProviderSecurityDomain" key access methods: decryptVerifyKey, openSecureChannel, unwrap, wrap, verifyExternalAuthenticate

The following SecureChannel key access methods unwrap, wrap, processSecurity , decryptData, encryptData, resetSecurity The following "APICrypto" key access methods: Key.clearKey, DES.getKey, DES.setKey, Signature.init, Signature.update, Signature.sign, Signature.verify, Cipher.init, Cipher.update, Cipher.doFinal AES DES

The following commands: PUT_KEY, STORE DATA, EXTERNAL AUTHENTICATE, INITIALIZE UPDATE, The following "ProviderSecurityDomain" key access methods: decryptVerifyKey, openSecureChannel, unwrap, verifyExternalAuthenticate

The following SecureChannel key access methods Unwrap, wrap, processSecurity, decryptData, encryptData, resetSecurity The following "APICrypto" key access methods: Key.clearKey, DES.getKey, DES.setKey, Signature.init, Signature.update, Signature.sign, Signature.verify, Cipher.init, Cipher.update, Cipher.doFinal RSA

The following "ProviderSecurityDomain" key access methods: DecryptVerifyKey,

The following "APICrypto" key access methods: Key.clearKey, RSAPrivateCRTKey.setP, RSAPrivateCRTKey.setQ, RSAPrivateCRTKey.setPQ, RSAPrivateCRTKey.setDP1, RSAPrivateCRTKey.setDQ1, RSAPrivateCRTKey.getP, RSAPrivateCRTKey.getQ, RSAPrivateCRTKey.getPQ, RSAPrivateCRTKey.getDP1, RSAPrivateCRTKey.getDQ1, RSAPrivateKey.setModulus, RSAPrivateKey.setExponent, RSAPrivateKey.getModulus, RSAPrivateKey.getExponent, RSAPublicKey.setModulus, RSAPublicKey.setExponent, RSAPublicKey.getModulus, RSAPublicKey.getExponent, Signature.init, Signature.update, Signature.sign, Signature.verify, Cipher.init, Cipher.update, Cipher KeyPair. genKeyPair doFinal ECkey.

The following "APICrypto" key access methods: Key.clearKey, ECPrivateKey.setFieldFP, ECPrivateKey.setA, ECPrivateKey.setB, ECPrivateKey.setG, ECPrivateKey.setR, ECPrivateKey.setK,ECPrivateKey.getField, ECPrivateKey.getA, ECPrivateKey.getB, ECPrivateKey.getG, ECPrivateKey.getR, ECPrivateKey.getK, ECPrivateKey.setS, ECPrivateKey.getS, ECPublicKey.setFieldFP, ECPublicKey.setA, ECPublicKey.setB, ECPublicKey.setG, ECPublicKey.setR,ECPublicKey.setK, ECPublicKey.getField, ECPublicKey.getA, ECPublicKey.getB, ECPublicKey.getG, ECPublicKey.getR,

ECPublicKey.getK, ECPublicKey.setW, ECPublicKey.getW, Signature.init,Signature.update, Signature.sign, Signature.verify KeyAgreement.init, KeyAgreement KeyPair. genKeyPair generateSecret.

FCS_CKM.4 Cryptographic key destruction

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method **The keys are reset in accordance with [6] in class Key with the method clearKey(). Any access to a cleared key attempting to use it for ciphering or signing shall throw an exception** that meets the following: **[6]**.

Application note:

That also prevents the destroyed keys from being referenced.

- The keys are reset as specified in [6] Key class, with the method clearKey(). Any access to a cleared key for ciphering or signing shall throw an exception.

FCS_COP.1 Cryptographic operation

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

Cryptographic operation	Cryptographic algorithm	key size	list of standard
signature, verification of signature, encryption and decryption	TDES	112 bits or 168 bits	FIPS 46-3 [16] FIPS 81 [17] **ISO 9797-1** [25]
signature, verification of signature, encryption and decryption	AES	from 128 to 256 bits with a step of 64 bits	**FIPS 197** [21]
signature, verification of signature, encryption and decryption	RSA	from 64 to 2048 bits with a step of 32-bit	1. Ansi X9.31 [11], 2. ISO / IEC 9796-1, annex A, section A.4 and A.5, and annex C [16] 3. PKCS#1 The public Key Cryptography standards, RSA Data Security Inc. 1993 [19]
Hash functions	SHA-1,SHA-224, SHA-256, SHA-384 and SHA-512	no keys	Secure Hash Standard, Federal Information Processing Standards Publication 180-3, 2008, october.
signature, verification of signature,	ECC	160, 192, 224, 256, 320, 384 and 512 bits	RFC 5639, ANSIx9.63, ANSIx9.62, FIPS186-3, IEEE Std 1363a-2004, SEC1, SEC2

Application note:

- The TOE shall provide a subset of cryptographic operations defined in [6] (see javacardx.crypto and javacard.security packages). Refinement of FCS_COP.1.1/OT/STMICROELECTRONICS/List of operations/list of algorithms/key sizes/list of standards

DES (IC)/OT has developed the algorithm using HW DES module/TDES encryption and decryption 56, 112 and 168 bits, FIPS 46-3 [16]

AES/OT has developed the algorithm/Data encryption / decryption//128/192/256 bits/FIPS 197 [21]

SHA1/OT has developed the algorithm/Hash function/SHA-1/No cryptographic key/FIPS 180-3 [19]

SHA224/OT has developed the algorithm/Hash function/SHA-224/No cryptographic key/ FIPS 180-3 [19]

SHA256/OT has developed the algorithm/Hash function/SHA-256/No cryptographic key/ FIPS 180-3 [19]

SHA384/OT has developed the algorithm/Hash function/SHA-384/No cryptographic key/ FIPS 180-3 [19]

SHA512/OT has developed the algorithm/Hash function/SHA-512/No cryptographic key/ FIPS 180-3 [19]

RSA without CRT /OT has developed the algorithm using HW PK accelerator/Data Encryption and Decryption/RSA Without CRT Data /Between 1024 bits to 2048 bits/PKCS#1 V2.0; 1st October, 1998 [26].

RSA with CRT /OT has developed the algorithm using HW PK accelerator/Data Encryption and Decryption/RSA With CRT Data /Between 1024 bits and 2048 bits/PKCS#1 V2.0; 1st October, 1998[26].

ECC/OT has developed the algorithm using HW PK accelerator/Signature and Verification and Key Exchange/160, 192, 224, 256, 320, 384 and 512 bits/ANSIx9.63, ANSIx9.62, FIPS186-3, IEEE Std 1363a-2000

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 [7], §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**.

Application note:

The allocation of a resource to the APDU buffer is typically performed as the result of a call to the process() method of an applet.

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 [6].

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 [6], §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 ([7], §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 [7], §7.7, within the bounds of the Commit Capacity ([7], §7.8), and those described in [6]**.

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 [6] (see for instance, PIN-blocking, PIN-checking, update of Transient objects).

Card Security Management

FAU_ARP.1 Security alarms

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

- **throw an exception,**
- **lock the card session,**
- **reinitialize the Java Card System and its data**

upon detection of a potential security violation.

Refinement:

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

- CAP file inconsistency,
- typing error in the operands of a bytecode,
- applet life cycle inconsistency,
- card tearing (unexpected removal of the Card out of the CAD) and power failure,
- abort of a transaction in an unexpected context, (see abortTransaction(), [6] and ([7], §7.6.2)
- violation of the Firewall or JCVM SFPs,
- unavailability of resources,
- array overflow,

FDP_SDI.2 Stored data integrity monitoring and action

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

FDP_SDI.2.2 Upon detection of a data integrity error, the TSF shall **increase counter of the Kill card file. If the maximum is reached (15) the the Kill card is launched**.

Application note:

The following data persistently stored by TOE have the user data attribute " integrityControlledData ": 1. PINs (i.e. objects instance of class OwnerPin or subclass of interface PIN) 2. keys (i.e. objects instance of classes implemented the interface Key) 3. SecureStores (i.e. objects instance of class SecureStore) 4. packages Java Card 5. patches

FPR_UNO.1 Unobservability

FPR_UNO.1.1 The TSF shall ensure that **any user** is unable to observe the operation **Cardholder authentication** on **D.PIN** by **user and no subject**.

FPT_FLS.1 Failure with preservation of secure state

FPT_FLS.1.1 The TSF shall preserve a secure state when the following types of failures occur: **those associated to the potential security violations described in 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 ([7], §6.2.3) or after a proximity card (PICC) activation sequence ([7]). Behavior of the TOE on power loss and reset is described in [7], §3.6 and §7.1. Behavior of the TOE on RF signal loss is described in [7], §3.6.1.

FPT_TDC.1 Inter-TSF basic TSF data consistency

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

FPT_TDC.1.2 The TSF shall use

- **the rules defined in [8] specification,**
- **the API tokens defined in the export files of reference implementation**

when interpreting the TSF data from another trusted IT product.

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 ([8], §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 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: **rules are defined in FDP_ACC.2/Firewall and ADP_ACF/Firewall**.

FIA_USB.1.3/AID The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: **none**.

Application note:

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

FMT_MTD.1/JCRE Management of TSF data

FMT_MTD.1.1/JCRE The TSF shall restrict the ability to **modify** the **list of registered applets' AIDs** to **the JCRE**.

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;

FMT_MTD.3/JCRE Secure TSF data

FMT_MTD.3.1/JCRE The TSF shall ensure that only secure values are accepted for **the registered applets' AIDs**.

7.1.2.2 InstG Security Functional Requirements

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

Package loading is allowed only if, for each dependent package, its AID attribute is equal to a resident package AID attribute, the major (minor) Version attribute associated to the dependent package is lesser than or equal to the major (minor) Version attribute associated to the resident package ([8], §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 [8] 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 ([8], §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 ([8], §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" ([8]) 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 ([8], §4.4).
- The installation (the invocation of an applet's install method by the installer) is implementation dependent ([7], §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, [7], §11).

FMT_SMR.1/Installer Security roles

FMT_SMR.1.1/Installer The TSF shall maintain the roles**: Installer**.

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

FPT_FLS.1/Installer Failure with preservation of secure state

FPT_FLS.1.1/Installer The TSF shall preserve a secure state when the following types of failures occur: **the installer fails to load/install a package/applet as described in [7] §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 **an applet (i.e. a package) is considered as loaded, once its reference is written in the list of the loaded packages (i.e. instantiated applets). This is the ultimate stage of the applet/package installation, done when everything has succeeded before (verification, initialization, object instantiation). If an error occurs before registration, everything must be rolled back. For package installation, the garbage collector will automatically remove the package code since we stopped installation before the package recording. For applet installation, we mainly relies on garbage collector, as it is done for package, to remove the applet instance and AID objects (since the applet is not on the root of persistence, these objects are unreachable). On applet installation, its install method is called which can lead to change the states of the VM objects. To rollback the modifications eventually made in field of other persistent objects, the installation is surrounded by a transaction (that is aborted). Finally, we have additional mechanisms to rollback modifications eventually done in the field of transient arrays since they are not covered but the transaction (volatile data is not in the scope of Java Card transaction)** is not possible, the TSF shall enter a maintenance mode where the ability to return to a secure state is provided.

FPT_RCV.3.2/Installer For **installation of the applet**, 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 **100%** 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.

7.1.2.3 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 [6].
- 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).

7.1.2.4 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 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 ([7], §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 ([7], §8.5) O.JAVAOBJECT is remote reachable.**
- **R.JAVA.15 ([7], §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 ([7], §8.5) O.JAVAOBJECT is remote reachable.**
- **R.JAVA.16 ([7], §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.**

o **R.JAVA.17 ([7], §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 ([7], §8.5) O.JAVAOBJECT is remote reachable**.

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

FDP_ACF.1.4/ADEL [Editorially Refined] The TSF shall explicitly deny access of **any subject but S.ADEL to O.CODE_PKG or O.APPLET for the purpose of deleting them from the card**.

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 protection profile.

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 [7], §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 initialization

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

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

FMT_SMF.1/ADEL Specification of Management Functions

FMT_SMF.1.1/ADEL The TSF shall be capable of performing the following management functions: **modify the list of registered applets' AIDs and the Resident Packages**.

Application note:

The modification of the Active Applets security attribute should be performed in accordance with the rules given in [7], §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 [7], §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 ([7], §11.3.4.)

7.1.2.5 CarG Security Functional Requirements

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

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

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

Application note:

In the TOE 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 a 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_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, 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.

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: **LoadFile, 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 de GP22 [9].**.

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

FDP_IFF.1.4/CM The TSF shall explicitly authorize 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 TOE fails to verify the integrity and authenticity evidences of the application packages.

Application note:

The protocol is well expressed in figure 9.2 and 9.3 of [9]

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.CMLIFECY t**o **CARD_Manager**.

FMT_MSA.3/CM Static attribute initialization

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

FMT_MSA.3.2/CM The TSF shall allow 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:

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

7.1.2.6 RMIG Security Functional Requirements

This group specifies the policies that control the access to the remote objects and the flow of information that takes place when the RMI service is used. The rules relate mainly to the lifetime of the remote references. Information concerning remote object references can be sent out of the card only if the corresponding remote object has been designated as exportable. Array parameters of remote method invocations must be allocated on the card as global arrays. Therefore, the storage of references to those arrays must be restricted as well. The JCRMI policy embodies both an access control and an information flow control policy.

The following objects are used in the RMI security requirements:

O.REMOTE_OBJ: A remote object is an instance of a class that implements one (or more) remote interfaces. A remote interface is one that extends, directly or indirectly, the interface java.rmi.Remote ([6]). O.REMOTE_MTHD: A method of a remote interface. O.RMI_SERVICE: These are instances of the class javacard.framework.service.RMIService. They are the objects that actually process the RMI services. O.ROR: A remote object reference. It provides information concerning: (i) the identification of a remote object and (ii) the Implementation class of the object or the interfaces implemented by the class of the object. This is the object's information to which the CAD can access.

FDP_ACC.2/JCRMI Complete access control

FDP_ACC.2.1/JCRMI The TSF shall enforce the **JCRMI access control SFP** on **S.CAD, S.JCRE, O.APPLET, O.REMOTE_OBJ, O.REMOTE_MTHD, O.ROR, O.RMI_SERVICE** and all operations among subjects and objects covered by the SFP.

Refinement:

The operations involved in this policy are:

- OP.GET_ROR,
- OP.INVOKE.

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

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

Subject/Object	Attributes
S.JCRE	Selected Applet Context
O.REMOTE_OBJ	Owner, Class, Identifier, ExportedInfo
O.REMOTE_MTHD	Identifier
O.RMI_SERVICE	Owner, Returned References

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

- **R.JAVA.18: S.CAD may perform OP.GET_ROR upon O.APPLET only if O.APPLET is the currently selected applet, and there exists an O.RMI_SERVICE with a registered initial reference to an O.REMOTE_OBJ that is owned by O.APPLET.**
- **R.JAVA.19: S.JCRE may perform OP.INVOKE upon O.RMI_SERVICE, O.ROR and O.REMOTE_MTHD only if O.ROR is valid (as defined in [7], §8.5) and it belongs to the Returned References of O.RMI_SERVICE, and if the Identifier of O.REMOTE_MTHD matches one of the remote methods in the Class of the O.REMOTE_OBJ to which O.ROR makes reference**.

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

FDP_ACF.1.4/JCRMI [Editorially Refined] The TSF shall explicitly deny access of **any subject but S.JCRE to O.REMOTE_OBJ and O.REMOTE_MTHD for the purpose of performing a remote method invocation**.

Application note:

FDP_ACF.1.2/JCRMI:

- The validity of a remote object reference is specified as a lifetime characterization. The security attributes involved in the rules for determining valid remote object references are the Returned References of the O.RMI_SERVICE and the Active Applets (see FMT_REV.1.1/JCRMI and FMT_REV.1.2/JCRMI). The precise mechanism by which a remote method is invoked on a remote object is defined in detail in ([7], §8.5.2 and [6]).
- Note that the owner of an O.RMI_SERVICE is the applet instance that created the object. The attribute Returned References lists the remote object references that have been sent to the S.CAD during the applet selection session. This attribute is implementation dependent.

FDP_IFC.1/JCRMI Subset information flow control

FDP_IFC.1.1/JCRMI The TSF shall enforce the **JCRMI information flow control SFP** on **S.JCRE, S.CAD, I.RORD and OP.RET_RORD(S.JCRE,S.CAD,I.RORD)**.

Application note:

FDP_IFC.1.1/JCRMI:

- Array parameters of remote method invocations must be allocated on the card as global arrays objects. References to global arrays cannot be stored in class variables, instance variables or array components. The control of the flow of that kind of information has already been specified in FDP_IFC.1.1/JCVM.
- A remote object reference descriptor is sent from the card to the CAD either as the result of a successful applet selection command ([7], §8.4.1), and in this case it describes, if any, the initial remote object reference of the selected applet; or as the result of a remote method invocation ([7],§8.3.5.1).

FDP_IFF.1/JCRMI Simple security attributes

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

Subjects/Information	Security attributes
I.RORD	ExportedInfo

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

OP.RET_RORD(S.JCRE, S.CAD, I.RORD) is permitted only if the attribute ExportedInfo of I.RORD has the value "true" ([7], §8.5).

FDP_IFF.1.3/JCRMI The TSF shall enforce the following rules: **none**.

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

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

Application note:

The ExportedInfo attribute of I.RORD indicates whether the O.REMOTE_OBJ which I.RORD identifies is exported or not (as indicated by the security attribute ExportedInfo of the O.REMOTE_OBJ).

FMT_MSA.1/EXPORT Management of security attributes

FMT_MSA.1.1/EXPORT The TSF shall enforce the **JCRMI access control SFP** to restrict the ability to **modify** the security attributes**: ExportedInfo of O.REMOTE_OBJ** to **its owner applet**.

Application note:

The Exported status of a remote object can be modified by invoking its methods export() and unexport(), and only the owner of the object may perform the invocation without raising a SecurityException (javacard.framework.service.CardRemoteObject). However, even if the owner of the object may provoke the change of the security attribute value, the modification itself can be performed by the Java Card RE.

FMT_MSA.1/REM_REFS Management of security attributes

FMT_MSA.1.1/REM_REFS The TSF shall enforce the **JCRMI access control SFP** to restrict the ability to **modify** the security attributes **Returned References of O.RMI_SERVICE** to **its owner applet**.

FMT_MSA.3/JCRMI Static attribute initialization

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

FMT_MSA.3.2/JCRMI 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.

Application note:

FMT_MSA.3.1/JCRMI:

- Remote objects' security attributes are created and initialized at the creation of the object, and except for the ExportedInfo attribute, the values of the attributes are not longer modifiable. The default value of the Exported attribute is true. There is one default value for the Selected Applet Context that is the default applet identifier's context, and one default value for the active context, that is "Java Card RE".

FMT_MSA.3.2/JCRMI:

- The intent is to have none of the identified roles to have privileges with regards 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.

FMT_REV.1/JCRMI Revocation

FMT_REV.1.1/JCRMI [Editorially Refined] The TSF shall restrict the ability to revoke **the Returned References of O.RMI_SERVICE** to **the Java Card RE**.

FMT_REV.1.2/JCRMI The TSF shall enforce the rules **that determine the lifetime of remote object references**.

Application note:

The rules are described in [7], §8.5

FMT_SMF.1/JCRMI Specification of Management Functions

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

- **modify the security attribute ExportedInfo of O.REMOTE_OBJ,**
- **modify the security attribute Returned References of O.RMI_SERVICE**.

FMT_SMR.1/JCRMI Security roles

FMT_SMR.1.1/JCRMI The TSF shall maintain the roles**: applet**.

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

Application note:

Applets own remote interface objects and may choose to allow or forbid their exportation, which is managed through a security attribute.

7.1.3 Functional requirements for the DESFire

The subjects defined here after are used in the specification of the Desfire SFRs:

. The Administrator i.e. the subject that owns or has access to the card master key.

. The Application Manager i.e. the subject that owns or has access to an application master key. Note that the TOE supports multiple applications and therefore multiple Application Managers, however for one application there is only one Application Manager.

. The Application User i.e. the subject that owns or has access to a key that allows to perform operations with application objects. Note that the TOE supports multiple Application Users within each application and the assigned rights to the Application Users can be different, which allows to have more or less powerful Application Users.

. Any other subject belongs to the role Everybody. This includes the card holder (i.e. end-user) and any other subject e.g. an attacker. These subjects do not possess any key and cannot perform operations that are restricted to the Administrator, Application Manager and Application User.

. The term Nobody will be used to explicitly indicate that no rights are granted to any subject.

The objects are defined in the sfrs are:

. The Card itself.

. The card can store a number of Applications.

. An application can store a number of Data Files of different types. One specific type of data file are Values.

Note that data files and values can be grouped in standard files and backup files, with values belonging to the group of backup files. When the term "file" is used without further information then both data files and values are meant.

The operations that can be performed with the objects in the desfire sfrs are:

-read a value or data from a data file,

-write data to a data file,

-increase a value (with a limit or unlimited),

-decrease a value,

-create an application, a value or a data file,

-delete an application, a value or a data file and

-modify attribute of the card, an application, a value or a data file. Note that 'freeze' will be used as specific form of modification that prevents any further modify.

The security attributes are

-Attributes of the card, applications, values and data files. There is a set of attributes for the card, a set of attributes for every application and a set of attributes for every single file within an application. The term "card attributes" will be used for the set of attributes related to the card, the term "application attributes" will be used for the set of application attributes and the term "file attributes" will be used for the attributes of values and data files.

Note that subjects are authorised by cryptographic keys. These keys are considered as authentication data and not as security attributes. The card has a card master key. Every application has an application master key and a variable number of keys used for operations on data files or values (all these keys are called application keys). The application keys within an application are numbered.

FMT_SMR.1/DF Security roles

FMT_SMR.1.1/DF The TSF shall maintain the roles**: *Administrator*, *Application Manager*, Application User and any authorized identified role.**

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

FDP_ACC.1/DF Subset access control

FDP_ACC.1.1/DF The TSF shall enforce the **Access Control Policy** on **all subjects, objects operations and attributes defined by the Access Control Policy.**

FDP_ACF.1/DF Security attributes based access control

FDP_ACF.1.1/DF The TSF shall enforce the **Access Control Policy** to objects based on the following: all subjects, objects and attributes.

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

The Administrator can create and delete applications

The Application Manager of an application can delete this application, create data file and values within this application and delete data files and values within this application.

An Application User can read or write a data file; read, increase or decrease a value based on the access control settings in the respective file attribute.

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

Everybody can create applications if this is allowed by a specific card attribute.

Everybody can create and delete data files or values of a specific application if this is allowed by a specific application attribute.

Everybody can read or write a data file; read, increase or decrease a value if this is allowed by a specific file attribute.

FDP_ACF.1.4/DF The TSF shall explicitly deny access of objects based on this rule:

Nobody can read or write a data file; read, increase or decrease a value if this is explicitly set for the respective operation on the respective data file or value.

FMT_MSA.3/DF Static attribute initialization

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

FMT_MSA.3.2/DF 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.

Application Note: The only initial attributes are the card attributes. All other attributes have to be defined at the same time the respective object is created.

FMT_MSA.1/DF Management of security attributes

FMT_MSA.1.1/DF The TSF shall enforce the **Access Control Policy** to restrict the ability to **modify or freeze** the security attributes **Card attributes, application attributes and file attributes** to **the Administrator, Application Manager and Application User, respectively.**

Refinement: The detailed management abilities are:

The Administrator can modify the card attributes. The card attributes contain a flag that when set will prevent any further change of the card attributes, thereby allowing to freeze the card attributes.

The Application Manager can modify the application attributes. The application attributes contain a flag that when set will prevent any further change of the application attributes, thereby allowing to freeze the application attributes.

The Application Manager can decide to restrict the ability to modify the file attributes to the Application Manager, an Application User, Everybody or to Nobody. The restriction to Nobody is equivalent to freezing the file attributes.

As an implication of the last rule, any subject that receives the modify abilities from the Application Manager gets these abilities transferred.

The implication given in the previous rule includes the possibility for an Application User to modify the file attributes if the Application Manager decides to transfer this ability. If there is no such explicit transfer an Application User does not have the ability to modify the file attributes.

FMT_SMF.1/DF Specification of Management Functions

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

Authenticate a user,

Invalidating the current authentication state based on the functions: Selecting an application or the card, Changing a key, Occurrence of any error during the execution of a command, Reset;

Changing a security attribute,

Creating or deleting an application, a value or a data file.

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

FDP_ITC.2.1/DF The TSF shall enforce the **Access Control Policy** when importing user data, controlled under the SFP, from outside of the TSC.

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

FDP_ITC.2.3/DF 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/DF 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/DF The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TSC: **no additional rules**.

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

FPT_TDC.1.1/DF The TSF shall provide the capability to consistently interpret **data files and values** when shared between the TSF and another trusted IT product.

FPT_TDC.1.2/DF The TSF shall use the rule: **data files or values can only be modified by their dedicated type-specific operations honouring the type-specific boundaries** when interpreting the TSF data from another trusted IT product.

Application Note: The TOE does not interpret the contents of the data, e.g. it cannot determine if data stored in a specific data file is an identification number that adheres to a specific format. Instead the TOE distinguishes different types of files and ensures that type-specific boundaries cannot be violated, e.g. values do not overflow, single records are limited by their size and cyclic records are handled correctly.

Implications of the Access Control Policy

The Access Control Policy has some implications, that can be drawn from the policy and that are essential parts of the TOE security functions.

The TOE end-user does normally not belong to the group of authorised users (Administrator, Application Manager, Application User), but regarded as ‘Everybody’ by the TOE. This means that the TOE cannot determine if it is used by its intended end-user (in other words: it cannot determine if the current card holder is the owner of the card).

The Administrator can have the exclusive right to create and delete applications on the Smart Card, however he can also grant this privilege to Everybody. Additionally, changing the Smart Card attributes is reserved for the Administrator. Application keys, at delivery time should be personalized to a preliminary, temporary key only known to the Administrator and the Application Manager.

At application personalization time, the Application Manager uses the preliminary application key in order to personalize the application keys, whereas all keys, except

the application master key, can be personalized to a preliminary, temporary key only known to the Application Manager and the Application User. Furthermore, the Application Manager has the right to create files within his application scope

FIA_UID.2/DF User identification before any action

FIA_UID.2.1/DF The TSF shall require each user to identify itself before allowing any other TSF-mediated actions on behalf of that user.

Application Note: Identification of a user is performed upon an authentication request based on the currently selected context and the key number. For example, if an authentication request for key number 0 is issued after selecting a specific application, the user is identified as the Application Manager of the respective application. Before any authentication request is issued the user is identified as 'Everybody'.

FIA_UAU.2/DF User authentication before any action

FIA_UAU.2.1/DF The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

FIA_UAU.5/DF Multiple authentication mechanisms

FIA_UAU.5.1/DF The TSF shall provide '**none' and cryptographic authentication** to support user authentication.

FIA_UAU.5.2/DF The TSF shall authenticate any user's claimed identity according to the following rules:

The 'none' authentication is performed with anyone who communicates with the TOE without issuing an explicit authentication request. The 'none' authentication implicitly and solely authorizes the 'Everybody' subject.

The cryptographic authentication is used to authorize the Administrator, Application Manager and Application User.

FMT_MTD.1/DF Management of TSF data

FMT_MTD.1.1 /DF The TSF shall restrict the ability to **change_default, modify or Freeze the card master key, application master keys and application keys to the Administrator, Application Manager and Application User**.

Refinement: The detailed management abilities are:

The Administrator can modify the card master key. The card attributes contains a flag that when set will prevent any further change of the card master key, thereby allowing to freeze the card master key.

The Administrator can change the default key that is used for the application master key and for the application keys when an application is created.

The Application Manager of an application can modify the application master key of this application. The application attributes contain a flag that when set will prevent any further change of the application master key, thereby allowing to freeze the application master key.

The Application Manager can decide to restrict the ability to modify the application keys to the Application Manager, the Application Users or to Nobody. The restriction to Nobody is equivalent to freezing the application keys. The Application Users can either change their own keys or one Application User can be defined that can change all keys of the Application Users within an application.

As an implication of the last rule, any subject that receives the modify abilities from the Application Manger gets these abilities transferred.

FTP_TRP.1/DF Trusted path

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

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

FTP_TRP.1.3/DF The TSF shall require the use of the trusted path for **authentication requests with DES and AES, confidentiality and/or data integrity verification for data transfers protected with AES and based on a setting in the file attributes.**

FDP_ROL.1/DF Basic rollback

FDP_ROL.1.1/DF The TSF shall enforce **Access Control Policy** to permit the rollback of the **operations that modify the value or data file objects** on the **backup files**.

FDP_ROL.1.2/DF The TSF shall permit operations to be rolled back within **the scope of the current transaction, which is defined by the following limitative events: chip reset, (re-)authentication (either successful or not), select command, explicit commit, explicit abort, command failure.**

FPT_RPL.1/DF Replay detection

FPT_RPL.1.1/DF The TSF shall detect replay for the following entities**: authentication requests with DES and AES, confidentiality and/or data integrity verification for data transfers protected with AES and based on a setting in the file attributes.**

FPT_RPL.1.2/DF The TSF shall perform **rejection of the request** when replay is detected.

7.1.4 Functional requirements for the IC

Security functions of the IC are all issued of the ST of IC INFINEON: SLE97CNFX1M50PE.

7.1.5 SCPG Security Functional Requirements

This group contains the security requirements for the smart card platform, that is, operating system and chip that the Java Card System is implemented upon. The requirements are expressed in terms of security functional requirements from [2]. The TSF and TSC stated in the following components refer to that of the SCP.

FPT_FLS.1/SCP Failure with preservation of secure state

FPT_FLS.1.1/SCP The TSF shall preserve a secure state when the following types of failures occur: **1. Invalid reference exception; 2. Code or data integrity failure; 3.**

Power loss while processing. 4. worm on or dead EEPROM, full security area, false CRC For each problem the TOE sends a specific exception status or doesn't start.

FRU_FLT.1/SCP Degraded fault tolerance

FRU_FLT.1.1/SCP The TSF shall ensure the operation of **Fault tolerance** when the following failures occur: **Lack of EEPROM**.

FPT_PHP.3/SCP Resistance to physical attack

FPT_PHP.3.1/SCP The TSF shall resist **changing operational conditions every times: the frequency of the external clock, power supply, and temperature** to the **chip elements by responding automatically such that the SFRs are always enforced** by responding automatically such that the SFRs are always enforced.

FPT_RCV.3/SCP Automated recovery without undue loss

FPT_RCV.3.1/SCP When automated recovery from **by power loss while reading from and writing to static and objects' fields** is not possible, the TSF shall enter a maintenance mode where the ability to return to a secure state is provided.

FPT_RCV.3.2/SCP For **all cases**, the TSF shall ensure the return of the TOE to a secure state using automated procedures.

FPT_RCV.3.3/SCP The functions provided by the TSF to recover from failure or service discontinuity shall ensure that the secure initial state is restored without exceeding **100%** for loss of TSF data or objects under the control of the TSF.

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

FPT_RCV.4/SCP Function recovery

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

FCS_RNG.1/SCP Random number generation

FCS_RNG.1.1/SCP The TSF shall provide a **hybrid Random Number generator** that implements: **none**.

FCS_RNG.1.2/SCP The TSF shall provide random numbers that meet **khi2 test**.

7.2 Security Assurance Requirements

The security assurance requirement level is EAL4 augmented with AVA_VAN.5 and ALC_DVS.2/Additional_refinement.

7.2.1 ADV Development

7.2.1.1 ADV_ARC Security Architecture

ADV_ARC.1 Security architecture description

ADV_ARC.1.1D The developer shall design and implement the TOE so that the security features of the TSF cannot be bypassed.

ADV_ARC.1.2D The developer shall design and implement the TSF so that it is able to protect itself from tampering by untrusted active entities.

ADV_ARC.1.3D The developer shall provide a security architecture description of the TSF.

ADV_ARC.1.1C The security architecture description shall be at a level of detail commensurate with the description of the SFR-enforcing abstractions described in the TOE design document.

ADV_ARC.1.2C The security architecture description shall describe the security domains maintained by the TSF consistently with the SFRs.

ADV_ARC.1.3C The security architecture description shall describe how the TSF initialization process is secure.

ADV_ARC.1.4C The security architecture description shall demonstrate that the TSF protects itself from tampering.

ADV_ARC.1.5C The security architecture description shall demonstrate that the TSF prevents bypass of the SFR-enforcing functionality.

ADV_ARC.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

7.2.1.2 ADV_FSP Functional specification

ADV_FSP.4 Complete functional specification

ADV_FSP.4.1D The developer shall provide a functional specification.

ADV_FSP.4.2D The developer shall provide a tracing from the functional specification to the SFRs.

ADV_FSP.4.1C The functional specification shall completely represent the TSF.

ADV_FSP.4.2C The functional specification shall describe the purpose and method of use for all TSFI.

ADV_FSP.4.3C The functional specification shall identify and describe all parameters associated with each TSFI.

ADV_FSP.4.4C The functional specification shall describe all actions associated with each TSFI.

ADV_FSP.4.5C The functional specification shall describe all direct error messages that may result from an invocation of each TSFI.

ADV_FSP.4.6C The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.

ADV_FSP.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.4.2E The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

7.2.1.3 ADV_IMP Implementation representation

ADV_IMP.1 Implementation representation of the TSF

ADV_IMP.1.1D The developer shall make available the implementation representation for the entire TSF.

ADV_IMP.1.2D The developer shall provide a mapping between the TOE design description and the sample of the implementation representation.

ADV_IMP.1.1C The implementation representation shall define the TSF to a level of detail such that the TSF can be generated without further design decisions.

ADV_IMP.1.2C The implementation representation shall be in the form used by the development personnel.

ADV_IMP.1.3C The mapping between the TOE design description and the sample of the implementation representation shall demonstrate their correspondence.

ADV_IMP.1.1E The evaluator shall confirm that, for the selected sample of the implementation representation, the information provided meets all requirements for content and presentation of evidence.

7.2.1.4 ADV_TDS TOE design

ADV_TDS.3 Basic modular design

ADV_TDS.3.1D The developer shall provide the design of the TOE.

ADV_TDS.3.2D The developer shall provide a mapping from the TSFI of the functional specification to the lowest level of decomposition available in the TOE design.

ADV_TDS.3.1C The design shall describe the structure of the TOE in terms of subsystems.

ADV_TDS.3.2C The design shall describe the TSF in terms of modules.

ADV_TDS.3.3C The design shall identify all subsystems of the TSF.

ADV_TDS.3.4C The design shall provide a description of each subsystem of the TSF.

ADV_TDS.3.5C The design shall provide a description of the interactions among all subsystems of the TSF.

ADV_TDS.3.6C The design shall provide a mapping from the subsystems of the TSF to the modules of the TSF.

ADV_TDS.3.7C The design shall describe each SFR-enforcing module in terms of its purpose and relationship with other modules.

ADV_TDS.3.8C The design shall describe each SFR-enforcing module in terms of its SFR-related interfaces, return values from those interfaces, interaction with other modules and called SFR-related interfaces to other SFR-enforcing modules.

ADV_TDS.3.9C The design shall describe each SFR-supporting or SFR-non-interfering module in terms of its purpose and interaction with other modules.

ADV_TDS.3.10C The mapping shall demonstrate that all TSFIs trace to the behavior described in the TOE design that they invoke.

ADV_TDS.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_TDS.3.2E The evaluator shall determine that the design is an accurate and complete instantiation of all security functional requirements.

7.2.2 AGD Guidance documents

7.2.2.1 AGD_OPE Operational user guidance

AGD_OPE.1 Operational user guidance

AGD_OPE.1.1D The developer shall provide operational user guidance.

AGD_OPE.1.1C The operational user guidance shall describe, for each user role, the user-accessible functions and privileges that should be controlled in a secure processing environment, including appropriate warnings.

AGD_OPE.1.2C The operational user guidance shall describe, for each user role, how to use the available interfaces provided by the TOE in a secure manner.

AGD_OPE.1.3C The operational user guidance shall describe, for each user role, the available functions and interfaces, in particular all security parameters under the control of the user, indicating secure values as appropriate.

AGD_OPE.1.4C The operational user guidance shall, for each user role, clearly present each type of security-relevant event relative to the user-accessible functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

AGD_OPE.1.5C The operational user guidance shall identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences and implications for maintaining secure operation.

AGD_OPE.1.6C The operational user guidance shall, for each user role, describe the security measures to be followed in order to fulfill the security objectives for the operational environment as described in the ST.

AGD_OPE.1.7C The operational user guidance shall be clear and reasonable.

AGD_OPE.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

7.2.2.2 AGD_PRE Preparative procedures

AGD_PRE.1 Preparative procedures

AGD_PRE.1.1D The developer shall provide the TOE including its preparative procedures.

AGD_PRE.1.1C The preparative procedures shall describe all the steps necessary for secure acceptance of the delivered TOE in accordance with the developer's delivery procedures.

AGD_PRE.1.2C The preparative procedures shall describe all the steps necessary for secure installation of the TOE and for the secure preparation of the operational environment in accordance with the security objectives for the operational environment as described in the ST.

AGD_PRE.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AGD_PRE.1.2E The evaluator shall apply the preparative procedures to confirm that the TOE can be prepared securely for operation.

7.2.3 ALC Life-cycle support

7.2.3.1 ALC_CMC CM capabilities

ALC_CMC.4 Production support, acceptance procedures and automation

ALC_CMC.4.1D The developer shall provide the TOE and a reference for the TOE.

ALC_CMC.4.2D The developer shall provide the CM documentation.

ALC_CMC.4.3D The developer shall use a CM system.

ALC_CMC.4.1C The TOE shall be labelled with its unique reference.

ALC_CMC.4.2C The CM documentation shall describe the method used to uniquely identify the configuration items.

ALC_CMC.4.3C The CM system shall uniquely identify all configuration items.

ALC_CMC.4.4C The CM system shall provide automated measures such that only authorized changes are made to the configuration items.

ALC_CMC.4.5C The CM system shall support the production of the TOE by automated means.

ALC_CMC.4.6C The CM documentation shall include a CM plan.

ALC_CMC.4.7C The CM plan shall describe how the CM system is used for the development of the TOE.

ALC_CMC.4.8C The CM plan shall describe the procedures used to accept modified or newly created configuration items as part of the TOE.

ALC_CMC.4.9C The evidence shall demonstrate that all configuration items are being maintained under the CM system.

ALC_CMC.4.10C The evidence shall demonstrate that the CM system is being operated in accordance with the CM plan.

ALC_CMC.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

7.2.3.2 ALC_CMS CM scope

ALC_CMS.4 Problem tracking CM coverage

ALC_CMS.4.1D The developer shall provide a configuration list for the TOE.

ALC_CMS.4.1C The configuration list shall include the following: the TOE itself; the evaluation evidence required by the SARs; the parts that comprise the TOE; the implementation representation; and security flaw reports and resolution status.

ALC_CMS.4.2C The configuration list shall uniquely identify the configuration items.

ALC_CMS.4.3C For each TSF relevant configuration item, the configuration list shall indicate the developer of the item.

ALC_CMS.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

7.2.3.3 ALC_DEL Delivery

ALC_DEL.1 Delivery procedures

ALC_DEL.1.1D The developer shall document and provide procedures for delivery of the TOE or parts of it to the consumer.

ALC_DEL.1.2D The developer shall use the delivery procedures.

ALC_DEL.1.1C The delivery documentation shall describe all procedures that are necessary to maintain security when distributing versions of the TOE to the consumer.

ALC_DEL.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

7.2.3.4 ALC_DVS Development security

ALC_DVS.2/Additional_refinement Sufficiency of security measures

ALC_DVS.2.1D/Additional_refinement The developer shall produce and provide development security documentation.

Refinement:

The TOE is developed at Infineon site and Oberthur sites. Oberthur sites are Colombes, Pessac and Jakarta, these sites are covered by audits.

The TOE is produced and personalized in Vitré, or in Shenzhen, Oberthur sites.

Security measures applied in this site are audited the audit includes the management of sensitive information and key management.

This audit concerns also the security of the keys to be loaded on the (U)SIM cards:

- Mobile operator keys including OTA keys (telecom keys either generated by the personalizer or by the mobile operator) and delegated management token keys,
- Issuer Security Domain keys (ISD keys or Card issuer keys),
- Application Provider Security Domains keys (APSD keys),
- Controlling Authority Security Domain keys (CASD keys),
- Verification Authority Security Domain keys (VASD keys).

The audit concerns also the application code before loading in pre-issuance, the organizational measures must be audited, the measures must ensure **that loaded application has not been changed since the code verifications** as required in OE.VERIFICATION (All the bytecodes shall be verified at least once, before the loading in order to ensure that each bytecode is valid at execution time).

ALC_DVS.2.1C/Additional_refinement The development security documentation shall describe all the physical, procedural, personnel, and other security measures that are necessary to protect the confidentiality and integrity of the TOE design and implementation in its development environment.

ALC_DVS.2.2C/Additional_refinement The development security documentation shall justify that the security measures provide the necessary level of protection to maintain the confidentiality and integrity of the TOE.

ALC_DVS.2.1E/Additional_refinement The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_DVS.2.2E/Additional_refinement The evaluator shall confirm that the security measures are being applied.

7.2.3.5 ALC_LCD Life-cycle definition

ALC_LCD.1 Developer defined life-cycle model

ALC_LCD.1.1D The developer shall establish a life-cycle model to be used in the development and maintenance of the TOE.

ALC_LCD.1.2D The developer shall provide life-cycle definition documentation.

ALC_LCD.1.1C The life-cycle definition documentation shall describe the model used to develop and maintain the TOE.

ALC_LCD.1.2C The life-cycle model shall provide for the necessary control over the development and maintenance of the TOE.

ALC_LCD.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

7.2.3.6 ALC_TAT Tools and techniques

ALC_TAT.1 Well-defined development tools

ALC_TAT.1.1D The developer shall provide the documentation identifying each development tool being used for the TOE.

ALC_TAT.1.2D The developer shall document and provide the selected implementation-dependent options of each development tool.

ALC_TAT.1.1C Each development tool used for implementation shall be well-defined.

ALC_TAT.1.2C The documentation of each development tool shall unambiguously define the meaning of all statements as well as all conventions and directives used in the implementation.

ALC_TAT.1.3C The documentation of each development tool shall unambiguously define the meaning of all implementation-dependent options.

ALC_TAT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_FLR.3 Systematic flaw remediation

ALC_FLR.3.1D The developer shall document and provide flaw remediation procedures addressed to TOE developers.

ALC_FLR.3.2D The developer shall establish a procedure for accepting and acting upon all reports of security flaws and requests for corrections to those flaws.

ALC_FLR.3.3D The developer shall provide flaw remediation guidance addressed to TOE users.

ALC_FLR.3.1C The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.

ALC_FLR.3.2C The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.

ALC_FLR.3.3C The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.

ALC_FLR.3.4C The flaw remediation procedures documentation shall describe the methods used to provide flaw information, corrections and guidance on corrective actions to TOE users.

ALC_FLR.3.5C The flaw remediation procedures shall describe a mean by which the developer receives from TOE users reports and enquiries of suspected security flaws in the TOE.

ALC_FLR.3.6C The flaw remediation procedures shall include a procedure requiring timely response and the automatic distribution of security flaw reports and the associated corrections to registered users who might be affected by the security flaw.

ALC_FLR.3.7C The procedures for processing reported security flaws shall ensure that any reported flaws are remediated and the remediation procedures issued to TOE users.

ALC_FLR.3.8C The procedures for processing reported security flaws shall provide safeguards that any corrections to these security flaws do not introduce any new flaws.

ALC_FLR.3.9C The flaw remediation guidance shall describe a means by which TOE users report to the developer any suspected security flaws in the TOE.

ALC_FLR.3.10C The flaw remediation guidance shall describe a means by which TOE users may register with the developer, to be eligible to receive security flaw reports and corrections.

ALC_FLR.3.11C The flaw remediation guidance shall identify the specific points of contact for all reports and enquiries about security issues involving the TOE.

ALC_FLR.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence

7.2.4 ASE Security Target evaluation

7.2.4.1 ASE_CCL Conformance claims

ASE_CCL.1 Conformance claims

ASE_CCL.1.1D The developer shall provide a conformance claim.

ASE_CCL.1.2D The developer shall provide a conformance claim rationale.

ASE_CCL.1.1C The conformance claim shall contain a CC conformance claim that identifies the version of the CC to which the ST and the TOE claim conformance.

ASE_CCL.1.2C The CC conformance claim shall describe the conformance of the ST to CC Part 2 as either CC Part 2 conformant or CC Part 2 extended.

ASE_CCL.1.3C The CC conformance claim shall describe the conformance of the ST to CC Part 3 as either CC Part 3 conformant or CC Part 3 extended.

ASE_CCL.1.4C The CC conformance claim shall be consistent with the extended components definition.

ASE_CCL.1.5C The conformance claim shall identify all PPs and security requirement packages to which the ST claims conformance.

ASE_CCL.1.6C The conformance claim shall describe any conformance of the ST to a package as either package-conformant or package-augmented.

ASE_CCL.1.7C The conformance claim rationale shall demonstrate that the TOE type is consistent with the TOE type in the PPs for which conformance is being claimed.

ASE_CCL.1.8C The conformance claim rationale shall demonstrate that the statement of the security problem definition is consistent with the statement of the security problem definition in the PPs for which conformance is being claimed.

ASE_CCL.1.9C The conformance claim rationale shall demonstrate that the statement of security objectives is consistent with the statement of security objectives in the PPs for which conformance is being claimed.

ASE_CCL.1.10C The conformance claim rationale shall demonstrate that the statement of security requirements is consistent with the statement of security requirements in the PPs for which conformance is being claimed.

ASE_CCL.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

7.2.4.2 ASE_ECD Extended components definition

ASE_ECD.1 Extended components definition

ASE_ECD.1.1D The developer shall provide a statement of security requirements.

ASE_ECD.1.2D The developer shall provide an extended components definition.

ASE_ECD.1.1C The statement of security requirements shall identify all extended security requirements.

ASE_ECD.1.2C The extended components definition shall define an extended component for each extended security requirement.

ASE_ECD.1.3C The extended components definition shall describe how each extended component is related to the existing CC components, families, and classes.

ASE_ECD.1.4C The extended components definition shall use the existing CC components, families, classes, and methodology as a model for presentation.

ASE_ECD.1.5C The extended components shall consist of measurable and objective elements such that conformance or nonconformance to these elements can be demonstrated.

ASE_ECD.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ASE_ECD.1.2E The evaluator shall confirm that no extended component can be clearly expressed using existing components.

7.2.4.3 ASE_INT ST introduction

ASE_INT.1 ST introduction

ASE_INT.1.1D The developer shall provide an ST introduction.

ASE_INT.1.1C The ST introduction shall contain an ST reference, a TOE reference, a TOE overview and a TOE description.

ASE_INT.1.2C The ST reference shall uniquely identify the ST.

ASE_INT.1.3C The TOE reference shall identify the TOE.

ASE_INT.1.4C The TOE overview shall summarize the usage and major security features of the TOE.

ASE_INT.1.5C The TOE overview shall identify the TOE type.

ASE_INT.1.6C The TOE overview shall identify any non-TOE hardware/software/firmware required by the TOE.

ASE_INT.1.7C The TOE description shall describe the physical scope of the TOE.

ASE_INT.1.8C The TOE description shall describe the logical scope of the TOE.

ASE_INT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ASE_INT.1.2E The evaluator shall confirm that the TOE reference, the TOE overview, and the TOE description are consistent with each other.

7.2.4.4 ASE_OBJ Security objectives

ASE_OBJ.2 Security objectives

ASE_OBJ.2.1D The developer shall provide a statement of security objectives.

ASE_OBJ.2.2D The developer shall provide security objectives rationale.

ASE_OBJ.2.1C The statement of security objectives shall describe the security objectives for the TOE and the security objectives for the operational environment.

ASE_OBJ.2.2C The security objectives rationale shall trace each security objective for the TOE back to threats countered by that security objective and OSPs enforced by that security objective.

ASE_OBJ.2.3C The security objectives rationale shall trace each security objective for the operational environment back to threats countered by that security objective, OSPs enforced by that security objective, and assumptions upheld by that security objective.

ASE_OBJ.2.4C The security objectives rationale shall demonstrate that the security objectives counter all threats.

ASE_OBJ.2.5C The security objectives rationale shall demonstrate that the security objectives enforce all OSPs.

ASE_OBJ.2.6C The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.

ASE_OBJ.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

7.2.4.5 ASE_REQ Security requirements

ASE_REQ.2 Derived security requirements

ASE_REQ.2.1D The developer shall provide a statement of security requirements.

ASE_REQ.2.2D The developer shall provide security requirements rationale.

ASE_REQ.2.1C The statement of security requirements shall describe the SFRs and the SARs.

ASE_REQ.2.2C All subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs shall be defined.

ASE_REQ.2.3C The statement of security requirements shall identify all operations on the security requirements.

ASE_REQ.2.4C All operations shall be performed correctly.

ASE_REQ.2.5C Each dependency of the security requirements shall either be satisfied, or the security requirements rationale shall justify the dependency not being satisfied.

ASE_REQ.2.6C The security requirements rationale shall trace each SFR back to the security objectives for the TOE.

ASE_REQ.2.7C The security requirements rationale shall demonstrate that the SFRs meet all security objectives for the TOE.

ASE_REQ.2.8C The security requirements rationale shall explain why the SARs were chosen.

ASE_REQ.2.9C The statement of security requirements shall be internally consistent.

ASE_REQ.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

7.2.4.6 ASE_SPD Security problem definition

ASE_SPD.1 Security problem definition

ASE_APD.1.1D The developer shall provide a security problem definition.

ASE_SPD.1.1C The security problem definition shall describe the threats.

ASE_SPD.1.2C All threats shall be described in terms of a threat agent, an asset, and an adverse action.

ASE_SPD.1.3C The security problem definition shall describe the OSPs.

ASE_SPD.1.4C The security problem definition shall describe the assumptions about the operational environment of the TOE.

ASE_SPD.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

7.2.4.7 ASE_TSS TOE summary specification

ASE_TSS.1 TOE summary specification

ASE_TSS.1.1D The developer shall provide a TOE summary specification.

ASE_TSS.1.1C The TOE summary specification shall describe how the TOE meets each SFR.

ASE_TSS.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ASE_TSS.1.2E The evaluator shall confirm that the TOE summary specification is consistent with the TOE overview and the TOE description.

7.2.5 ATE Tests

7.2.5.1 ATE_COV Coverage

ATE_COV.2 Analysis of coverage

ATE_COV.2.1D The developer shall provide an analysis of the test coverage.

ATE_COV.2.1C The analysis of the test coverage shall demonstrate the correspondence between the tests in the test documentation and the TSFIs in the functional specification.

ATE_COV.2.2C The analysis of the test coverage shall demonstrate that all TSFIs in the functional specification have been tested.

ATE_COV.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

7.2.5.2 ATE_DPT Depth

ATE_DPT.1 Testing: basic design

ATE_DPT.1.1D The developer shall provide the analysis of the depth of testing.

ATE_DPT.1.1C The analysis of the depth of testing shall demonstrate the correspondence between the tests in the test documentation and the TSF subsystems in the TOE design.

ATE_DPT.1.2C The analysis of the depth of testing shall demonstrate that all TSF subsystems in the TOE design have been tested.

ATE_DPT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

7.2.5.3 ATE_FUN Functional tests

ATE_FUN.1 Functional testing

ATE_FUN.1.1D The developer shall test the TSF and document the results.

ATE_FUN.1.2D The developer shall provide test documentation.

ATE_FUN.1.1C The test documentation shall consist of test plans, expected test results and actual test results.

ATE_FUN.1.2C The test plans shall identify the tests to be performed and describe the scenarios for performing each test. These scenarios shall include any ordering dependencies on the results of other tests.

ATE_FUN.1.3C The expected test results shall show the anticipated outputs from a successful execution of the tests.

ATE_FUN.1.4C The actual test results shall be consistent with the expected test results.

ATE_FUN.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

7.2.5.4 ATE_IND Independent testing

ATE_IND.2 Independent testing - sample

ATE_IND.2.1D The developer shall provide the TOE for testing.

ATE_IND.2.1C The TOE shall be suitable for testing.

ATE_IND.2.2C The developer shall provide an equivalent set of resources to those that were used in the developer's functional testing of the TSF.

ATE_IND.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_IND.2.2E The evaluator shall execute a sample of tests in the test documentation to verify the developer test results.

ATE_IND.2.3E The evaluator shall test a subset of the TSF to confirm that the TSF operates as specified.

7.2.6 AVA Vulnerability assessment

7.2.6.1 AVA_VAN Vulnerability analysis

AVA_VAN.5 Advanced methodical vulnerability analysis

AVA_VAN.5.1D The developer shall provide the TOE for testing.

AVA_VAN.5.1C The TOE shall be suitable for testing.

AVA_VAN.5.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VAN.5.2E The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the TOE.

AVA_VAN.5.3E The evaluator shall perform an independent, methodical vulnerability analysis of the TOE using the guidance documentation, functional specification, TOE design, security architecture description and implementation representation to identify potential vulnerabilities in the TOE.

AVA_VAN.5.4E The evaluator shall conduct penetration testing based on the identified potential vulnerabilities to determine that the TOE is resistant to attacks performed by an attacker possessing High attack potential.

7.3 Security Requirements Rationale

7.3.1 Objectives

7.3.1.1 Security Objectives for the TOE

(U)SIM part

Card Management

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

- FDP_UIT.1/CCM enforces the Secure Channel Protocol information flow control policy and the Security Domain access control policy to ensure the integrity of card management operations.
- FDP_ROL.1/CCM ensures that card management operations may be cleanly aborted.
- FDP_ITC.2/CCM enforces the Firewall access control policy and the Secure Channel Protocol information flow policy when importing card management data.
- FPT_TDC.1/CCM-OT enforces the consistency of security attributes when transmitted to a Security Domain.
- FPT_FLS.1/CCM preserves a secure state when failures occur.
- All SFRs related to Security Domains (FDP_ACC.1/SD, FDP_ACF.1/SD, FMT_MSA.1/SD, FMT_MSA.3/SD, FMT_SMF.1/SD, FMT_SMR.1/SD) cover this

security objective by enforcing a Security Domain access control policy (rules and restrictions) that ensures a secure card content management.

- All SFRs related to the secure channel (FMT_MSA.1/SC, FMT_MSA.3/SC, FMT_SMF.1/SC, FIA_UAU.1/SC, FTP_ITC.1/SC, FCO_NRO.2/SC, FDP_IFC.2/SC, FDP_IFF.1/SC, FIA_UID.1/SC, FIA_UAU.4/SC, FIA_UAU.7/SC-OT, FIA_AFL.1/SC-OT) support this security objective by enforcing Secure Channel Protocol information flow control policy that ensures the integrity and the authenticity of card management operations.
- FPR_UNO.1/SC-OT enforces the unobservability of the imported keys. All the 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. All security requirements from group 'CarG Security Functional Requirements' contributes to cover this security objective by preventing the installation of packages that has not been bytecode verified, or that has been modified after bytecode verification. FMT_SMR.1,FPT_FLS.1,FDP_IFC.2/CM,FCO_NRO.2/CM,FDP_IFF.1/CM,FIA_UID.1/CM, FMT_MSA.1/CM, FMT_MSA.3/CM, FMT_SMF.1/CM, FMT_SMR.1/CM, FTP_ITC.1/CM.

O.DOMAIN-RIGHTS The security objective O.DOMAIN-RIGHTS is met by the following SFRs:

- All SFRs related to Security Domains (FDP_ACC.1/SD, FDP_ACF.1/SD, FMT_MSA.1/SD, FMT_MSA.3/SD, FMT_SMF.1/SD, FMT_SMR.1/SD) cover this security objective by enforcing a Security Domain access control policy (rules and restrictions) that ensures a secure card content management.
- All SFRs related to the secure channel (FMT_MSA.1/SC, FMT_MSA.3/SC, FMT_SMF.1/SC, FIA_UAU.1/SC, FTP_ITC.1/SC, FCO_NRO.2/SC, FDP_IFC.2/SC, FDP_IFF.1/SC, FIA_UID.1/SC, FIA_UAU.4/SC) support this security objective by enforcing Secure Channel Protocol information flow control policy that ensures the integrity and the authenticity of card management operations.

O.APPLI-AUTH The security objective O.APPLI-AUTH is met by the following SFRs:

- FDP_ROL.1/CCM ensures that card management operations may be cleanly aborted.
- FPT_FLS.1/CCM preserves a secure state when failures occur.
- FCS_COP.1/DAP ensures that the loaded Executable Application is legitimate by specifying the algorithm to be used in order to verify the DAP signature of the Verification Authority.
- FCS_COP.1/TOKEN-OT_TDES, FCS_COP.1/TOKEN-OT_AES and FCS_COP.1/TOKEN-OT_RSA ensures that the card content management command is authorized by verifying the Token signature.

- FCS_COP.1/RECEIPTS-OT(TDES, AES) ensures that the card content management command has been successfully processed by computing the Receipt signature.

Communication

O.COMM_AUTH This security objective is covered by the following security functional requirements:

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

The security functional requirement cryptographic operation: FCS_COP.1/DAP, FCS_COP.1/TOKEN-OT (TDES, AES and RSA) , FCS_COP.1/RECEIPT-OT(TDES, AES), FCS_COP.1/CIPHERLOADFILE-OT (TDES and AES), defined in [7] supports also this security objective by specifying secure cryptographic algorithm that shall be used to determine the origin of the card management commands.

O.COMM_INTEGRITY This security objective is covered by the following security functional requirements:

- FTP_ITC.1/SC which ensures the integrity of card management commands.
- FMT_SMF.1/SC specifies the actions activating the integrity check on the card management commands.
- FMT_SMR.1/SD defines the roles enabling to send and authenticate the card management requests for which the integrity has to be ensured.
- FDP_IFC.2/SC and FDP_IFF.1/SC enforces the Secure Channel Protocol information flow control policy to guarantee the integrity of administration requests.
- FMT_MSA.1/SC and FMT_MSA.3/SC covers indirectly this security objective by specifying security attributes enabling to guarantee the integrity of card management requests.

The security functional requirement defined in [7] supports also this security objective by specifying secure cryptographic algorithm that shall be used to ensure the integrity of the card management commands: FCS_COP.1/DAP, FCS_COP.1/TOKEN-OT (TDES, AES and RSA), FCS_COP.1/RECEIPTS-OT(TDES, RSA and AES), FCS_COP.1/CIPHERLOADFILE-OT (TDES and AES).

O.COMM_CONFIDENTIALITY This security objective is covered by the following security functional requirements:

- FTP_ITC.1/SC which ensures the confidentiality of card management commands.
- FMT_SMF.1/SC specifies the actions ensuring the confidentiality of the card management commands.
- FMT_SMR.1/SD defines the roles enabling to send and authenticate the card management requests for which the confidentiality has to be ensured.

- o FDP_IFC.2/SC and FDP_IFF.1/SC enforces the Secure Channel Protocol information flow control policy to guarantee the confidentiality of administration requests.
- o FMT_MSA.1/SC and FMT_MSA.3/SC covers indirectly this security objective by specifying security attributes enabling to guarantee the confidentiality of card management requests by decrypting those requests and imposing management conditions on that attributes.
- o FCS_COP.1/CIPHERLOADFILE-OT (TDES and AES), enforces the decryption of the Ciphered Load File Data Block in accordance with a specified cryptographic algorithm.

Java Card System Protection Profile - Open Configuration

IDENTIFICATION

O.SID < Open > 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/REM_REFS, FMT_MSA.1/EXPORT, FMT_MSA.1/ADEL, FMT_MSA.1/CM, FMT_MSA.3/JCRMI, FMT_MSA.3/ADEL, FMT_MSA.3/FIREWALL, FMT_MSA.3/JCVM, FMT_MSA.3/CM, FMT_SMF.1/CM, FMT_SMF.1/ADEL, FMT_SMF.1/ADEL, FMT_SMF.1/JCRMI, FMT_MTD.1/JCRE and FMT_MTD.3/JCRE FMT_SMF.1/SD,FMT_SMF.1/SC,FMT_SMF.1.

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 JCRMI access control policy (FDP_ACC.2/JCRMI, FDP_ACF.1/JCRMI) and the functional requirement FDP_ITC.2/Installer. The functional requirements of the class FMT (FMT_MTD.1/JCRE, FMT_MTD.3/JCRE, FMT_SMR.1/Installer, FMT_SMR.1, FMT_SMF.1, FMT_SMR.1/ADEL, FMT_SMR.1/JCRMI, FMT_SMF.1/ADEL, FMT_SMF.1/JCRMI, FMT_SMF.1/CM, FMT_MSA.1/CM, FMT_MSA.3/CM, FMT_SMR.1/CM, FMT_MSA.2/FIREWALL_JCVM, FMT_MSA.3/FIREWALL, FMT_MSA.3/JCVM, FMT_MSA.1/ADEL, FMT_MSA.3/ADEL, FMT_MSA.1/EXPORT, FMT_MSA.1/REM_REFS, FMT_MSA.3/JCRMI, FMT_MSA.1/JCRE, FMT_MSA.1/JCVM, FMT_REV.1/JCRMI) also indirectly contribute to meet this objective.

O.GLOBAL_ARRAYS_CONFID Only arrays can be designated as global, and the only global arrays required in the Java Card API are the APDU buffer and the global byte array input parameter (bArray) to an applet's install method. The clearing requirement of these arrays is met by (FDP_RIP.1/APDU 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.

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.APPLET, which uphold the assumption A.APPLET.

O.OPERATE The TOE is protected in various ways against applets' actions (FPT_TDC.1), the FIREWALL access control policy FDP_ACC.2/FIREWALL and FDP_ACF.1/FIREWALL, and is able to detect and block various failures or security violations during usual working (FPT_FLS.1/ADEL, FPT_FLS.1, FPT_FLS.1/ODEL, FPT_FLS.1/Installer, FAU_ARP.1). Its security-critical parts and procedures are also protected: safe recovery from failure is ensured (FPT_RCV.3/Installer), applets' installation may be cleanly aborted (FDP_ROL.1/FIREWALL), communication with external users and their internal subjects is well-controlled (FDP_ITC.2/Installer, FIA_ATD.1/AID, FIA_USB.1/AID) to prevent alteration of TSF data (also protected by components of the FPT class).

Almost every objective and/or functional requirement indirectly contributes to this one too.

Application note: Startup of the TOE (TSF-testing) can be covered by FPT_TST.1. This SFR component is not mandatory in [7], but appears in most of security requirements documents for masked applications. Testing could also occur randomly. Self-tests may become mandatory in order to comply to FIPS certification [FIPS 140-2].

O.REALLOCATION This security objective is satisfied by the following SFRs: FDP_RIP.1/APDU, FDP_RIP.1/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/TRANSIENT, FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/ADEL, which imposes that the contents of the re-allocated block shall always be cleared before delivering the block.

O.RESOURCES The TSFs detects stack/memory overflows during execution of applications (FAU_ARP.1, FPT_FLS.1/ADEL, FPT_FLS.1, FPT_FLS.1/ODEL, FPT_FLS.1/Installer). Failed installations are not to create memory leaks (FDP_ROL.1/FIREWALL, FPT_RCV.3/Installer) as well. Memory management is controlled by the TSF (FMT_MTD.1/JCRE, FMT_MTD.3/JCRE, FMT_SMR.1/Installer, FMT_SMR.1, FMT_SMF.1 FMT_SMR.1/ADEL, FMT_SMR.1/JCRMI, FMT_SMF.1/ADEL, FMT_SMF.1/JCRMI, FMT_SMF.1/CM and FMT_SMR.1/CM).

SERVICES

O.ALARM This security objective is met by FPT_FLS.1/Installer, FPT_FLS.1, FPT_FLS.1/ADEL, FPT_FLS.1/ODEL which guarantee that a secure state is preserved by the TSF when failures occur, and FAU_ARP.1 which defines TSF reaction upon detection of a potential security violation.

O.CIPHER This security objective is directly covered by FCS_CKM.1, FCS_CKM.2, FCS_CKM.3, FCS_CKM.4 and FCS_COP.1. The SFR FPR_UNO.1 contributes in covering this security objective and controls the observation of the cryptographic operations which may be used to disclose the keys.

O.KEY-MNGT This relies on the same security functional requirements as O.CIPHER, plus FDP_RIP.1 and FDP_SDI.2 as well. Precisely it is met by the following components: FCS_CKM.1, FCS_CKM.2, FCS_CKM.3, FCS_CKM.4, FCS_COP.1, 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/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/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL, FDP_RIP.1/TRANSIENT, FPR_UNO.1, FDP_ROL.1/FIREWALL and FDP_SDI.2 security functional requirements. The TSFs behind these are implemented by API classes. The firewall security functions FDP_ACC.2/FIREWALL and FDP_ACF.1/FIREWALL shall protect the access to private and internal data of the objects.

O.TRANSACTION Directly met by FDP_ROL.1/FIREWALL, FDP_RIP.1/ABORT, FDP_RIP.1/ODEL, FDP_RIP.1/APDU, FDP_RIP.1/bArray, FDP_RIP.1/KEYS, FDP_RIP.1/ADEL, FDP_RIP.1/TRANSIENT and FDP_RIP.1/OBJECTS (more precisely, by the element FDP_RIP.1.1/ABORT).

O.REMOTE The access to the TOE's internal data and the flow of information from the card to the CAD required by the JCRMI service is under control of the JCRMI access control

policy (FDP_ACC.2/JCRMI, FDP_ACF.1/JCRMI) and the JCRMI information flow control policy (FDP_IFC.1/JCRMI, FDP_IFF.1/JCRMI). The security functional requirements of the class FMT (FMT_MSA.1/EXPORT, FMT_MSA.1/REM_REFS, FMT_MSA.3/JCRMI, FMT_REV.1/JCRMI and FMT_SMR.1/JCRMI) contribute to meet this objective.

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).

Objectives on the TOE for the DESFire

O.DF_Access-Control: The SFR FMT_SMR.1/DF defines the roles of the Access Control Policy. The SFR FDP_ACC.1/DF and FDP_ACF.1/DF define the rules and FMT_MSA.3/DF and FMT_MSA.1/DF the attributes that the access control is based on.

FMT_MTD.1/DF provides the rules for the management of the authentication data. The management functions are defined by FMT_SMF.1/DF.

Since the TOE stores data on behalf of the authorised subjects import of user data with security attributes is defined by FDP_ITC.2/DF. Since cryptographic keys are used for authentication (refer to O.DF_Authentication), these keys have to be removed if they are no longer needed for the access control (i.e. an application is deleted). This is required by FCS_CKM.4.

These SFR together provide an access control mechanism as required by the objective O.DF_Access-Control.

O.DF_Authentication

The cryptographic SFRs (FCS_COP.1 for TDES (with 112 bits and 168 bits) and FCS_COP.1 for AES (from 128 bits to 256 bits)) require that the TOE provides the basic cryptographic algorithms that can be used to perform the authentication. The SFR FIA_UID.2/DF, FIA_UAU.2/DF and FIA_UAU.5/DF together define that users must be identified and authenticated before any action. The 'none' authentication of FIA_UAU.5/DF also ensures that a specific subject is identified and authenticated before an explicit authentication request is sent to the TOE. FTP_TRP.1/DF requires a trusted communication path between the TOE and remote users, FTP_TRP.1.3/DF especially requires "authentication requests". Together with FPT_RPL.1/DF which requires a replay detection for these authentication requests the seven SFR fulfill the objective O.DF_Authentication.

O.DF_Confidentiality

The SFR FCS_COP.1 (AES from 128 bits to 256 bits) requires that the TOE provides the basic cryptographic algorithm that can be used to protect the communication by encryption. FTP_TRP.1/DF requires a trusted communication path between the TOE and remote users,
FTP_TRP.1.3/DF especially requires "confidentiality and/or data integrity verification for data transfers protected and based on a setting in the file attributes". Together with FPT_RPL.1/DF which requires replay detection for these data transfers.

O.DF_Type-Consistency

The SFR FPT_TDC.1/DF requires the TOE to consistently interpret data files and values. The TOE will honour the respective file formats and boundaries (i.e. upper and lower limits, size limitations). This meets the objective O.DF_Type-Consistency.

O.DF_Transaction

The SFR FDP_ROL.1/DF requires the possibility to rollback a set of modifying operations on backup files in total. The set of operations is defined by the scope of the transaction, which is itself limited by some boundary events.

Additional Objectives on the TOE

O.SCP.SUPPORT The components FPT_RCV.3/SCP and FPT_RCV.4/SCP 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. The component FRU_FLT.1/SCP contributes to O.SCP.SUPPORT by preventing an incoherency of the written data. The component FPT_PHP.3/SCP contributes to O.SCP.SUPPORT by resisting to abnormal environmental conditions. The component FCS_RNG.1/SCP supports secure low-level RNG.

O.SCP.IC This objective is met by the component FPT_PHP.3/SCP the component shall resist changing operational conditions every times.

O.SCP.RECOVERY The component FPT_RCV.3/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/SCP, FPT_RCV.3/SCP and FRU_FLT.1/SCP.

7.3.2 Rationale tables of Security Objectives and SFRs

Security Objectives	Security Functional Requirements	Rationale
O.CARD-MANAGEMENT	FMT_MSA.3/SD, FMT_SMR.1/SD, FDP_UIT.1/CCM, FDP_ROL.1/CCM, FDP_ITC.2/CCM, FPT_FLS.1/CCM, FMT_MSA.1/SC, FMT_MSA.3/SC, FMT_SMF.1/SC, FIA_UAU.1/SC, FTP_ITC.1/SC, FCO_NRO.2/SC, FDP_IFC.2/SC, FDP_IFF.1/SC, FIA_UID.1/SC, FIA_UAU.4/SC, FDP_ACF.1/SD, FMT_MSA.1/SD, FMT_SMF.1/SD, FIA_AFL.1/SC-OT, FIA_UAU.7/SC-OT, FPR_UNO.1/SC-OT, FPT_RCV.3/Installer, FMT_SMR.1, FPT_FLS.1, FDP_IFC.2/CM, FCO_NRO.2/CM, FDP_IFF.1/CM, FIA_UID.1/CM, FMT_MSA.1/CM, FMT_MSA.3/CM, FMT_SMF.1/CM, FMT_SMR.1/CM, FTP_ITC.1/CM, 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_ACC.1/SD, FPT_TDC.1/CCM-OT	Section 6.3.1

Security Objectives	Security Functional Requirements	Rationale
O.DOMAIN-RIGHTS	FDP_ACF.1/SD, FMT_MSA.3/SD, FMT_SMR.1/SD, FMT_MSA.1/SC, FMT_MSA.3/SC, FMT_SMF.1/SC, FIA_UID.1/SC, FIA_UAU.1/SC, FDP_IFC.2/SC, FDP_IFF.1/SC, FMT_MSA.1/SD, FMT_SMF.1/SD, FTP_ITC.1/SC, FCO_NRO.2/SC, FIA_UAU.4/SC, FDP_ACC.1/SD	Section 6.3.1
O.APPLI-AUTH	FDP_ROL.1/CCM, FPT_FLS.1/CCM, FCS_COP.1/DAP, FCS_COP.1/TOKEN-OT (TDES, AES and RSA), FCS_COP.1/RECEIPTS-OT(TDES and AES)	Section 6.3.1
O.COMM_AUTH	FMT_SMR.1/SD, FDP_IFC.2/SC, FDP_IFF.1/SC, FMT_MSA.1/SC, FMT_MSA.3/SC, FIA_UID.1/SC, FIA_UAU.1/SC, FTP_ITC.1/SC, FCS_COP.1/DAP, FCS_COP.1/TOKEN-OT(TDES, AES and RSA), FCS_COP.1/RECEIPTS-OT(TDES, AES), FCS_COP.1/CIPHERLOADFILE-OT (TDES and AES)	Section 6.3.1
O.COMM_INTEGRITY	FMT_SMF.1/SC, FMT_SMR.1/SD, FDP_IFC.2/SC, FDP_IFF.1/SC, FMT_MSA.1/SC, FMT_MSA.3/SC, FTP_ITC.1/SC, FCS_COP.1/TOKEN-OT (TDES, AES and RSA), FCS_COP.1/RECEIPTS-OT(TDES, AES), FCS_COP.1/CIPHERLOADFILE-OT (TDES and AES), FCS_COP.1/DAP	Section 6.3.1
O.COMM_CONFIDENTIALITY	FMT_SMF.1/SC, FMT_SMR.1/SD, FDP_IFC.2/SC, FDP_IFF.1/SC, FMT_MSA.1/SC, FMT_MSA.3/SC, FCS_COP.1/CIPHERLOADFILE-OT (TDES and AES), FTP_ITC.1/SC	Section 6.3.1
O.SID	FIA_ATD.1/AID, FIA_UID.2/AID, FMT_MSA.1/JCRE, FMT_MSA.3/FIREWALL, FMT_MTD.1/JCRE, FMT_MTD.3/JCRE, FIA_USB.1/AID, FMT_MSA.1/JCVM, FMT_MSA.3/JCVM, FDP_ITC.2/Installer, FMT_SMF.1/SD, FMT_SMF.1/SC, FMT_SMF.1, FMT_MSA.1/ADEL, FMT_MSA.3/ADEL, FMT_SMF.1/ADEL, FMT_MSA.1/REM_REFS, FMT_MSA.3/JCRMI, FMT_SMF.1/JCRMI, FMT_MSA.1/EXPORT, FMT_MSA.1/CM, FMT_MSA.3/CM, FMT_SMF.1/CM	Section 6.3.1

Security Objectives	Security Functional Requirements	Rationale
O.FIREWALL	FDP_IFC.1/JCVM, FDP_IFF.1/JCVM, FMT_MSA.3/FIREWALL, FMT_SMR.1, FMT_MSA.1/JCRE, FDP_ACC.2/FIREWALL, FDP_ACF.1/FIREWALL, FMT_SMF.1, FMT_MSA.2/FIREWALL_JCVM, FMT_MTD.1/JCRE, FMT_MTD.3/JCRE, FMT_MSA.1/JCVM, FMT_MSA.3/JCVM, FDP_ITC.2/Installer, FMT_SMR.1/Installer, FMT_MSA.1/ADEL, FMT_MSA.3/ADEL, FMT_SMR.1/ADEL, FMT_SMF.1/ADEL, FDP_ACC.2/JCRMI, FDP_ACF.1/JCRMI, FMT_MSA.1/REM_REFS, FMT_REV.1/JCRMI, FMT_MSA.3/JCRMI, FMT_SMF.1/JCRMI, FMT_SMR.1/JCRMI, FMT_MSA.1/EXPORT, FMT_MSA.1/CM, FMT_MSA.3/CM, FMT_SMF.1/CM, FMT_SMR.1/CM	Section 6.3.1
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/TRANSIENT, FDP_RIP.1/ADEL	Section 6.3.1
O.GLOBAL_ARRAYS_INTEG	FDP_IFC.1/JCVM, FDP_IFF.1/JCVM	Section 6.3.1
O.NATIVE	FDP_ACF.1/FIREWALL	Section 6.3.1
O.OPERATE	FAU_ARP.1, FDP_ROL.1/FIREWALL, FIA_ATD.1/AID, FPT_FLS.1, FPT_FLS.1/ODEL, FDP_ACC.2/FIREWALL, FDP_ACF.1/FIREWALL, FPT_TDC.1, FIA_USB.1/AID, FDP_ITC.2/Installer, FPT_FLS.1/Installer, FPT_RCV.3/Installer, FPT_FLS.1/ADEL	Section 6.3.1
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/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/ADEL	Section 6.3.1
O.RESOURCES	FAU_ARP.1, FDP_ROL.1/FIREWALL, FMT_SMR.1, FPT_FLS.1/ODEL, FPT_FLS.1, FMT_SMF.1, FMT_MTD.1/JCRE, FMT_MTD.3/JCRE, FMT_SMR.1/Installer, FPT_FLS.1/Installer, FPT_RCV.3/Installer, FMT_SMR.1/ADEL, FPT_FLS.1/ADEL, FMT_SMF.1/ADEL, FMT_SMF.1/JCRMI, FMT_SMR.1/JCRMI, FMT_SMF.1/CM, FMT_SMR.1/CM	Section 6.3.1
O.ALARM	FPT_FLS.1, FPT_FLS.1/ODEL, FAU_ARP.1, FPT_FLS.1/Installer, FPT_FLS.1/ADEL	Section 6.3.1

Security Objectives	Security Functional Requirements	Rationale
O.CIPHER	FCS_CKM.1, FCS_CKM.2, FCS_CKM.3, FCS_CKM.4, FCS_COP.1, FPR_UNO.1	Section 6.3.1
O.KEY-MNGT	FCS_CKM.1, FCS_CKM.2, FCS_CKM.3, FCS_CKM.4, FCS_COP.1, 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_SDI.2, FDP_RIP.1/TRANSIENT, FDP_RIP.1/ADEL	Section 6.3.1
O.PIN-MNGT	FDP_RIP.1/ODEL, FDP_RIP.1/OBJECTS, FDP_RIP.1/APDU, FDP_RIP.1/bArray, FDP_RIP.1/ABORT, FDP_RIP.1/KEYS, FPR_UNO.1, FDP_RIP.1/TRANSIENT, FDP_ROL.1/FIREWALL, FDP_SDI.2, FDP_ACC.2/FIREWALL, FDP_ACF.1/FIREWALL, FDP_RIP.1/ADEL	Section 6.3.1
O.TRANSACTION	FDP_ROL.1/FIREWALL, FDP_RIP.1/ABORT, FDP_RIP.1/ODEL, FDP_RIP.1/APDU, FDP_RIP.1/bArray, FDP_RIP.1/KEYS, FDP_RIP.1/TRANSIENT, FDP_RIP.1/OBJECTS, FDP_RIP.1/ADEL	Section 6.3.1
O.REMOTE	FDP_ACC.2/JCRMI, FDP_ACF.1/JCRMI, FDP_IFC.1/JCRMI, FDP_IFF.1/JCRMI, FMT_MSA.1/EXPORT, FMT_MSA.1/REM_REFS, FMT_MSA.3/JCRMI, FMT_REV.1/JCRMI, FMT_SMR.1/JCRMI	Section 6.3.1
O.OBJ-DELETION	FDP_RIP.1/ODEL, FPT_FLS.1/ODEL	Section 6.3.1
O.DELETION	FDP_ACC.2/ADEL, FDP_ACF.1/ADEL, FDP_RIP.1/ADEL, FPT_FLS.1/ADEL, FMT_MSA.1/ADEL, FMT_MSA.3/ADEL, FMT_SMR.1/ADEL, FPT_RCV.3/Installer	Section 6.3.1
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 6.3.1
O.INSTALL	FDP_ITC.2/Installer, FPT_RCV.3/Installer, FPT_FLS.1/Installer	Section 6.3.1
O.SCP.SUPPORT	FPT_RCV.3/SCP, FPT_RCV.4/SCP, FCS_RNG.1/SCP, FPT_PHP.3/SCP, FRU_FLT.1/SCP	Section 6.3.1
O.SCP.IC	FPT_PHP.3/SCP	Section 6.3.1
O.SCP.RECOVERY	FPT_FLS.1/SCP, FRU_FLT.1/SCP, FPT_RCV.3/SCP	Section 6.3.1

Security Objectives	Security Functional Requirements	Rationale
O.DF_Access-Control	SFR FMT_SMR.1/DF, SFR FDP_ACC.1/DF FDP_ACF.1/DF, FMT_MSA.3/DF, FMT_MSA.1/DF, FMT_MTD.1/DF, FMT_SMF.1/DF, FDP_ITC.2/DF., FCS_CKM.4.	Section 6.3.1
O.DF_Authentication	FCS_COP.1, FIA_UID.2/DF,FIA_UAU.2/DF, FIA_UAU.5/DF,FIA_UAU.5/DF, FTP_TRP.1/DF, FTP_TRP.1.3/DF, FPT_RPL.1/DF	Section 6.3.1
O.DF_Confidentiality	FCS_COP.1, FTP_TRP.1/DF, FTP_TRP.1.3/DF, FPT_RPL.1/DF	Section 6.3.1
O.DF_Type-Consistency	FPT_TDC.1/DF	Section 6.3.1
O.DF_Transaction	FDP_ROL.1/DF	Section 6.3.1

Table 15 Security Objectives and SFRs - Coverage

Security Functional Requirements	Security Objectives
FDP_UIT.1/CCM	O.CARD-MANAGEMENT
FDP_ROL.1/CCM	O.CARD-MANAGEMENT, O.APPLI-AUTH
FDP_ITC.2/CCM	O.CARD-MANAGEMENT
FPT_FLS.1/CCM	O.CARD-MANAGEMENT, O.APPLI-AUTH
FCS_COP.1/DAP	O.APPLI-AUTH, O.COMM_AUTH, O.COMM_INTEGRITY
FCS_COP.1/TOKEN-OT(TDES, AES and RSA)	O.APPLI-AUTH, O.COMM_AUTH, O.COMM_INTEGRITY
FCS_COP.1/RECEIPTS-OT(TDES, AES)	O.APPLI-AUTH, O.COMM_AUTH, O.COMM_INTEGRITY
FCS_COP.1/CIPHERLOADFILE-OT(TDES and AES)	O.COMM_AUTH, O.COMM_INTEGRITY, O.COMM_CONFIDENTIALITY
FPT_TDC.1/CCM-OT	O.CARD-MANAGEMENT
FDP_ACF.1/SD	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS
FMT_SMR.1/SD	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS, O.COMM_AUTH, O.COMM_INTEGRITY, O.COMM_CONFIDENTIALITY
FMT_MSA.3/SD	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS
FMT_MSA.1/SD	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS
FMT_SMF.1/SD	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS, O.SID
FDP_ACC.1/SD	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS
FTP_ITC.1/SC	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS, O.COMM_AUTH, O.COMM_INTEGRITY, O.COMM_CONFIDENTIALITY
FCO_NRO.2/SC	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS
FDP_IFC.2/SC	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS, O.COMM_AUTH, O.COMM_INTEGRITY, O.COMM_CONFIDENTIALITY
FDP_IFF.1/SC	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS, O.COMM_AUTH, O.COMM_INTEGRITY, O.COMM_CONFIDENTIALITY
FMT_MSA.3/SC	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS, O.COMM_AUTH, O.COMM_INTEGRITY, O.COMM_CONFIDENTIALITY
FMT_SMF.1/SC	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS, O.COMM_INTEGRITY, O.COMM_CONFIDENTIALITY, O.SID
FIA_UID.1/SC	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS, O.COMM_AUTH

Security Functional Requirements	Security Objectives
FIA_UAU.1/SC	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS, O.COMM_AUTH
FIA_UAU.4/SC	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS
FIA_AFL.1/SC-OT	O.CARD-MANAGEMENT
FIA_UAU.7/SC-OT	O.CARD-MANAGEMENT
FPR_UNO.1/SC-OT	O.CARD-MANAGEMENT
FMT_MSA.1/SC	O.CARD-MANAGEMENT, O.DOMAIN-RIGHTS, O.COMM_AUTH, O.COMM_INTEGRITY, O.COMM_CONFIDENTIALITY
FDP_ACC.2/FIREWALL	O.FIREWALL, O.OPERATE, O.PIN-MNGT
FDP_ACF.1/FIREWALL	O.FIREWALL, O.NATIVE, O.OPERATE, O.PIN-MNGT
FDP_IFC.1/JCVM	O.FIREWALL, O.GLOBAL_ARRAYS_CONFID, O.GLOBAL_ARRAYS_INTEG
FDP_IFF.1/JCVM	O.FIREWALL, O.GLOBAL_ARRAYS_CONFID, O.GLOBAL_ARRAYS_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	O.SID, O.FIREWALL, O.RESOURCES
FMT_SMR.1	O.CARD-MANAGEMENT, O.FIREWALL, O.RESOURCES
FCS_CKM.1	O.CIPHER, O.KEY-MNGT
FCS_CKM.2	O.CIPHER, O.KEY-MNGT
FCS_CKM.3	O.CIPHER, O.KEY-MNGT
FCS_CKM.4	O.CIPHER, O.KEY-MNGT O.DF_Access-Control
FCS_COP.1	O.CIPHER, O.KEY-MNGT O.DF_Confidentiality O.DF_Authentication
FDP_RIP.1/ABORT	O.GLOBAL_ARRAYS_CONFID, O.REALLOCATION, O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION

Security Functional Requirements	Security Objectives
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.OPERATE, O.RESOURCES, O.PIN-MNGT, O.TRANSACTION
FAU_ARP.1	O.OPERATE, O.RESOURCES, O.ALARM
FDP_SDI.2	O.KEY-MNGT, O.PIN-MNGT
FPR_UNO.1	O.CIPHER, O.KEY-MNGT, O.PIN-MNGT
FPT_FLS.1	O.CARD-MANAGEMENT, O.OPERATE, O.RESOURCES, O.ALARM
FPT_TDC.1	O.OPERATE
FIA_ATD.1/AID	O.SID, O.OPERATE
FIA_UID.2/AID	O.SID
FIA_USB.1/AID	O.SID, O.OPERATE
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.OPERATE, O.INSTALL
FMT_SMR.1/Installer	O.FIREWALL, O.RESOURCES
FPT_FLS.1/Installer	O.OPERATE, O.RESOURCES, O.ALARM, O.INSTALL
FPT_RCV.3/Installer	O.CARD-MANAGEMENT, O.OPERATE, O.RESOURCES, O.DELETION, O.INSTALL
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.OPERATE, O.RESOURCES, O.ALARM, O.OBJ-DELETION
FDP_ACC.2/ADEL	O.CARD-MANAGEMENT, O.DELETION
FDP_ACF.1/ADEL	O.CARD-MANAGEMENT, O.DELETION

Security Functional Requirements	Security Objectives
FDP_RIP.1/ADEL	O.CARD-MANAGEMENT, O.GLOBAL_ARRAYS_CONFID, O.REALLOCATION, O.KEY-MNGT, O.PIN-MNGT, O.TRANSACTION, O.DELETION
FMT_MSA.1/ADEL	O.CARD-MANAGEMENT, O.SID, O.FIREWALL, O.DELETION
FMT_MSA.3/ADEL	O.CARD-MANAGEMENT, O.SID, O.FIREWALL, O.DELETION
FMT_SMF.1/ADEL	O.SID, O.FIREWALL, O.RESOURCES
FMT_SMR.1/ADEL	O.CARD-MANAGEMENT, O.FIREWALL, O.RESOURCES, O.DELETION
FPT_FLS.1/ADEL	O.CARD-MANAGEMENT, O.OPERATE, O.RESOURCES, O.ALARM, O.DELETION
FCO_NRO.2/CM	O.CARD-MANAGEMENT, O.LOAD
FDP_IFC.2/CM	O.CARD-MANAGEMENT, O.LOAD
FDP_UIT.1/CM	O.LOAD
FDP_IFF.1/CM	O.CARD-MANAGEMENT, O.LOAD
FIA_UID.1/CM	O.CARD-MANAGEMENT, O.LOAD
FMT_MSA.1/CM	O.CARD-MANAGEMENT, O.SID, O.FIREWALL
FMT_MSA.3/CM	O.CARD-MANAGEMENT, O.SID, O.FIREWALL
FMT_SMF.1/CM	O.CARD-MANAGEMENT, O.SID, O.FIREWALL, O.RESOURCES
FMT_SMR.1/CM	O.CARD-MANAGEMENT, O.FIREWALL, O.RESOURCES
FTP_ITC.1/CM	O.CARD-MANAGEMENT, O.LOAD
FDP_ACC.2/JCRMI	O.FIREWALL, O.REMOTE
FDP_ACF.1/JCRMI	O.FIREWALL, O.REMOTE
FDP_IFC.1/JCRMI	O.REMOTE
FDP_IFF.1/JCRMI	O.REMOTE
FMT_MSA.1/EXPORT	O.SID, O.FIREWALL, O.REMOTE
FMT_MSA.1/REM_REFS	O.SID, O.FIREWALL, O.REMOTE
FMT_MSA.3/JCRMI	O.SID, O.FIREWALL, O.REMOTE
FMT_REV.1/JCRMI	O.FIREWALL, O.REMOTE
FMT_SMF.1/JCRMI	O.SID, O.FIREWALL, O.RESOURCES
FMT_SMR.1/JCRMI	O.FIREWALL, O.RESOURCES, O.REMOTE
FPT_FLS.1/SCP	O.SCP.RECOVERY
FRU_FLT.1/SCP	O.SCP.SUPPORT, O.SCP.RECOVERY

Security Functional Requirements	Security Objectives
FPT_PHP.3/SCP	O.SCP.SUPPORT, O.SCP.IC
FPT_RCV.3/SCP	O.SCP.SUPPORT, O.SCP.RECOVERY
FPT_RCV.4/SCP	O.SCP.SUPPORT
FCS_RNG.1/SCP	O.SCP.SUPPORT
FMT_SMR.1/DF	O.DF_Access-Control
FMT_MSA.1/DF	O.DF_Access-Control
FDP_ACC.1/DF	O.DF_Access-Control
FDP_ACF.1/DF	O.DF_Access-Control
FDP_ROL.1/DF	O.DF_Transaction
FMT_MSA.3/DF	O.DF_Access-Control
FMT_MTD.1/DF	O.DF_Access-Control
FMT_SMF.1/DF	O.DF_Access-Control
FDP_ITC.2/DF	O.DF_Access-Control
FIA_UID.2/DF	O.DF_Authentication
FIA_UAU.2/DF	O.DF_Authentication
FIA_UAU.5/DF	O.DF_Authentication
FTP_TRP.1/DF	O.DF_Authentication O.DF_Confidentiality
FPT_RPL.1/DF	O.DF_Authentication O.DF_Confidentiality
FPT_TDC.1/DF	O.DF_Type-Consistency

Table 16: SFRs and Security Objectives

7.3.3 Dependencies

7.3.3.1 SFRs dependencies

Requirements	CC Dependencies	Satisfied Dependencies
FPT_FLS.1/SCP	No dependencies	
FRU_FLT.1/SCP	(FPT_FLS.1)	FPT_FLS.1/SCP
FPT_PHP.3/SCP	No dependencies	
FPT_RCV.3/SCP	(AGD_OPE.1)	AGD_OPE.1
FPT_RCV.4/SCP	No dependencies	
FCS_RNG.1/SCP	No dependencies	
FDP_ITC.2/Installer	(FDP_ACC.1 or FDP_IFC.1) and (FPT_TDC.1) and (FTP_ITC.1 or FTP_TRP.1)	FDP_IFC.2/CM, FTP_ITC.1/CM, FPT_TDC.1
FMT_SMR.1/Installer	(FIA_UID.1)	
FPT_FLS.1/Installer	No dependencies	
FPT_RCV.3/Installer	(AGD_OPE.1)	AGD_OPE.1
FDP_RIP.1/ODEL	No dependencies	
FPT_FLS.1/ODEL	No dependencies	
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	
FCO_NRO.2/CM	(FIA_UID.1)	FIA_UID.1/CM
FDP_IFC.2/CM	(FDP_IFF.1)	FDP_IFF.1/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
FDP_IFF.1/CM	(FDP_IFC.1) and (FMT_MSA.3)	FDP_IFC.2/CM, FMT_MSA.3/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

Requirements	CC Dependencies	Satisfied Dependencies
FMT_MSA.3/CM	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/CM, FMT_SMR.1/CM
FMT_SMF.1/CM	No dependencies	
FMT_SMR.1/CM	(FIA_UID.1)	FIA_UID.1/CM
FTP_ITC.1/CM	No dependencies	
FDP_ACC.2/JCRMI	(FDP_ACF.1)	FDP_ACF.1/JCRMI
FDP_ACF.1/JCRMI	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.2/JCRMI, FMT_MSA.3/JCRMI
FDP_IFC.1/JCRMI	(FDP_IFF.1)	FDP_IFF.1/JCRMI
FDP_IFF.1/JCRMI	(FDP_IFC.1) and (FMT_MSA.3)	FDP_IFC.1/JCRMI, FMT_MSA.3/JCRMI
FMT_MSA.1/EXPORT	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.2/JCRMI, FMT_SMF.1/JCRMI, FMT_SMR.1/JCRMI
FMT_MSA.1/REM_REFS	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.2/JCRMI, FMT_SMF.1/JCRMI, FMT_SMR.1/JCRMI
FMT_MSA.3/JCRMI	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/EXPORT, FMT_MSA.1/REM_REFS, FMT_SMR.1/JCRMI
FMT_REV.1/JCRMI	(FMT_SMR.1)	FMT_SMR.1/JCRMI
FMT_SMF.1/JCRMI	No dependencies	
FMT_SMR.1/JCRMI	(FIA_UID.1)	FIA_UID.2/AID
FDP_UIT.1/CCM	(FDP_ACC.1 or FDP_IFC.1) and (FTP_ITC.1 or FTP_TRP.1)	FDP_ACC.1/SD, FTP_ITC.1/SC
FDP_ROL.1/CCM	(FDP_ACC.1 or FDP_IFC.1)	FDP_ACC.1/SD
FDP_ITC.2/CCM	(FDP_ACC.1 or FDP_IFC.1) and (FPT_TDC.1) and (FTP_ITC.1 or FTP_TRP.1)	FPT_TDC.1/CCM-OT, FDP_ACC.1/SD, FTP_ITC.1/SC
FPT_FLS.1/CCM	No dependencies	
FCS_COP.1/DAP	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FDP_ITC.2/CCM
FCS_COP.1/TOKEN-OT(TDES, AES and RSA)	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FDP_ITC.2/CCM
FCS_COP.1/RECEIPTS-OT(TDES, AES)	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FDP_ITC.2/CCM
FCS_COP.1/CIPHERLOADFILE-OT(TDES and AES)	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FDP_ITC.2/CCM
FPT_TDC.1/CCM-OT	No dependencies	

Requirements	CC Dependencies	Satisfied Dependencies
FDP_ACF.1/SD	(FDP_ACC.1) and (FMT_MSA.3)	FMT_MSA.3/SD, FDP_ACC.1/SD
FMT_SMR.1/SD	(FIA_UID.1)	FIA_UID.1/SC
FMT_MSA.3/SD	(FMT_MSA.1) and (FMT_SMR.1)	FMT_SMR.1/SD, FMT_MSA.1/SD
FMT_MSA.1/SD	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FMT_SMR.1/SD, FMT_SMF.1/SD, FDP_ACC.1/SD
FMT_SMF.1/SD	No dependencies	
FDP_ACC.1/SD	(FDP_ACF.1)	FDP_ACF.1/SD
FTP_ITC.1/SC	No dependencies	
FCO_NRO.2/SC	(FIA_UID.1)	FIA_UID.1/SC
FDP_IFC.2/SC	(FDP_IFF.1)	FDP_IFF.1/SC
FDP_IFF.1/SC	(FDP_IFC.1) and (FMT_MSA.3)	FDP_IFC.2/SC, FMT_MSA.3/SC
FMT_MSA.3/SC	(FMT_MSA.1) and (FMT_SMR.1)	FMT_SMR.1/SD, FMT_MSA.1/SC
FMT_SMF.1/SC	No dependencies	
FIA_UID.1/SC	No dependencies	
FIA_UAU.1/SC	(FIA_UID.1)	FIA_UID.1/SC
FIA_UAU.4/SC	No dependencies	
FIA_AFL.1/SC-OT	(FIA_UAU.1)	FIA_UAU.1/SC
FIA_UAU.7/SC-OT	(FIA_UAU.1)	FIA_UAU.1/SC
FPR_UNO.1/SC-OT	No dependencies	
FMT_MSA.1/SC	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FMT_SMR.1/SD, FDP_ACC.1/SD, FMT_SMF.1/SC
FDP_ACC.2/FIREWALL	(FDP_ACF.1)	FDP_ACF.1/FIREWALL
FDP_ACF.1/FIREWALL	(FDP_ACC.1) and (FMT_MSA.3)	FDP_ACC.2/FIREWALL, FMT_MSA.3/FIREWALL
FDP_IFC.1/JCVM	(FDP_IFF.1)	FDP_IFF.1/JCVM
FDP_IFF.1/JCVM	(FDP_IFC.1) and (FMT_MSA.3)	FDP_IFC.1/JCVM, FMT_MSA.3/JCVM
FDP_RIP.1/OBJECTS	No dependencies	
FMT_MSA.1/JCRE	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.2/FIREWALL, FMT_SMR.1

Requirements	CC Dependencies	Satisfied Dependencies
FMT_MSA.1/JCVM	(FDP_ACC.1 or FDP_IFC.1) and (FMT_SMF.1) and (FMT_SMR.1)	FDP_ACC.2/FIREWALL, FDP_IFC.1/JCVM, FMT_SMF.1, FMT_SMR.1
FMT_MSA.2/FIREWALL_JCVM	(FDP_ACC.1 or FDP_IFC.1) and (FMT_MSA.1) and (FMT_SMR.1)	FDP_ACC.2/FIREWALL, FDP_IFC.1/JCVM, FMT_MSA.1/JCRE, FMT_MSA.1/JCVM, FMT_SMR.1
FMT_MSA.3/FIREWALL	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/JCRE, FMT_MSA.1/JCVM, FMT_SMR.1
FMT_MSA.3/JCVM	(FMT_MSA.1) and (FMT_SMR.1)	FMT_MSA.1/JCVM, FMT_SMR.1
FMT_SMF.1	No dependencies	
FMT_SMR.1	(FIA_UID.1)	FIA_UID.2/AID
FCS_CKM.1	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_CKM.2, FCS_CKM.4
FCS_CKM.2	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1, FCS_CKM.4
FCS_CKM.3	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1, FCS_CKM.4
FCS_CKM.4	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2)	FCS_CKM.1
FCS_COP.1	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1, FCS_CKM.4
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
FAU_ARP.1	(FAU_SAA.1)	
FDP_SDI.2	No dependencies	
FPR_UNO.1	No dependencies	
FPT_FLS.1	No dependencies	
FPT_TDC.1	No dependencies	
FIA_ATD.1/AID	No dependencies	
FIA_UID.2/AID	No dependencies	

Requirements	CC Dependencies	Satisfied Dependencies
FIA_USB.1/AID	(FIA_ATD.1)	FIA_ATD.1/AID
FMT_MTD.1/JCRE	(FMT_SMF.1) and (FMT_SMR.1)	FMT_SMF.1, FMT_SMR.1
FMT_MTD.3/JCRE	(FMT_MTD.1)	FMT_MTD.1/JCRE

Table 17: SFRs dependencies

Requirements	CC Dependencies	Satisfied Dependencies
FMT_SMR.1/DF	FIA_UID.2	FIA_UID.2/DF
FDP_ACC.1/DF	FDP_ACF.1	FDP_ACF.1/DF
FDP_ACF.1/DF	FDP_ACC.1 FMT_MSA.3	FDP_ACC.1/DF FMT_MSA.3/DF
FMT_MSA.3/DF	FMT_MSA.1 FMT_SMR.1	FMT_MSA.1/DF FMT_SMR.1/DF
FMT_MSA.1/DF	FDP_ACC.1 or FDP_IFC.1 And FMT_SMR.1 And FMT_SMF.1	FDP_ACC.1/DF FMT_SMR.1/DF FMT_SMF.1/DF
FMT_MTD.1/DF	FMT_SMF.1, FMT_SMR.1	FMT_SMF.1/DF, FMT_SMR.1/DF
FMT_SMF.1/DF	No dependencies	
FDP_ITC.2/DF	FDP_ACC.1 or FDP_IFC.1 and FTP_ITC.1 or FTP_TRP.1 and FPT_TDC.1	FDP_ACC.1/DF FTP_TRP.1/DF FPT_TDC.1/DF
FIA_UID.2/DF	No dependencies	
FIA_UAU.2/DF	FIA_UID.1	FIA_UID.2/DF
FIA_UAU.5/DF	No dependencies	
FTP_TRP.1/DF	No dependencies	
FPT_RPL.1/DF	No dependencies	
FPT_TDC.1/DF	No dependencies	
FDP_ROL.1/DF	FDP_ACC.1 or FDP_IFC.1	FDP_ACF.1/DF

All the dependencies of Desfire SFrs are supported.

Table 18: SFRs dependencies for DesFire

Rationale for the exclusion of dependencies

The dependency FIA_UID.1 of FMT_SMR.1/Installer is unsupported. This PP does not require the identification of the "installer" since it can be considered as part of the TSF.

The dependency FIA_UID.1 of FMT_SMR.1/ADEL is unsupported. This PP 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 unsupported. The dependency between FMT_MSA.1/JCRE and FMT_SMF.1 is not satisfied because no management functions are required for the Java Card RE.

The dependency FAU_SAA.1 of FAU_ARP.1 is unsupported. The dependency of FAU_ARP.1 on FAU_SAA.1 assumes that a "potential security violation" generates an audit event. On the contrary, the events listed in FAU_ARP.1 are self-contained (arithmetic exception, ill-formed bytecodes, access failure) and ask for a straightforward reaction of the TSFs on their occurrence at runtime. The JCVM or other components of the TOE detect these events during their usual working order. Thus, there is no mandatory audit recording in this PP.

7.3.3.2 SARs dependencies

Requirements	CC Dependencies	Satisfied Dependencies
ADV_ARC.1	(ADV_FSP.1) and (ADV_TDS.1)	ADV_FSP.4, ADV_TDS.3
ADV_FSP.4	(ADV_TDS.1)	ADV_TDS.3
ADV_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/Additional_refinement, ALC_LCD.1
ALC_CMS.4	No dependencies	
ALC_DEL.1	No dependencies	
ALC_FLR.3	No dependencies	
ALC_DVS.2/Additional_refinement	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	

Requirements	CC Dependencies	Satisfied 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 19: SARs dependencies

7.3.4 Rationale for the Security Assurance Requirements

EAL4 allows a developer to attain a reasonably high assurance level without the need for highly specialized processes and practices. It corresponds to a white box analysis and it can be considered as a reasonable level that can be applied to an existing product line without undue expense and complexity.

The TOE is intended to operate in open environments, where attackers can easily exploit vulnerabilities. According to the claimed intended usage of the TOE, it is very likely that it may represent a significant value and then constitute an attractive target for attacks. In some malicious usages of the TOE the statistical or probabilistic mechanisms in the TOE, for instance, may be subjected to analysis and attack in the normal course of operation. An EAL 4 augmented with ALC_DVS.2 and AVA_VAN.5 seems to be the reasonable minimum level for (U) SIM cards hosting sensitive applications. It shall probably be the case, as it is frequent nowadays, that the required evaluation assurance level will be high in, for instance, banking or electronic signature applications.

7.3.5 AVA_VAN.5 Advanced methodical vulnerability analysis

This component is added to EAL 4 package in order to provide sufficient robustness to counter an attacker with high attack potential without the support of a protecting environment.

Moreover, the (U)SIM card is a generic platform that could be used for a wide range of applications, including highly sensitive ones, like identity cards, pay-TV, e-cards, or credit cards. Potential attackers for such kind of applications include international organizations, or even a state, disposing of important means and resources.

7.3.6 ALC_DVS.2/Additional_refinement Sufficiency of security measures

This component is added in order to provide a higher assurance on the security of the (U)SIM cards development and manufacturing processes, especially for the secure handling of the embedded software and data.

Those requirements appear as the most adequate ones for a manufacturing process in which several actors (Platform Developer, Operator, Application Developers, IC Manufacturer, etc) exchange and store highly sensitive information (confidential code, cryptographic keys, personalization data, etc).

This assurance requirement will be evaluated for development and the personalization environment.

The personalizer is in charge of the TOE personalization process before card issuance. He ensures the security of the keys he loads on the (U)SIM cards:

- Mobile operator keys including OTA keys (telecom keys either generated by the personalizer or by the mobile operator) and delegated management token keys,
- Issuer Security Domain keys (ISD keys or Card issuer keys),
- Application Provider Security Domains keys (APSD keys),
- Controlling Authority Security Domain keys (CASD keys),
- Verification Authority Security Domain keys (VASD keys).

The personalization can rely either on the key escrow if the APSD has been created before the usage phase of the (U)SIM card or on the CA if the APSD has been created during the usage phase.

The security of all the keys must be ensured by a well defined security policy that covers generation, storage, distribution, destruction and recovery. This policy is audited.

7.3.7 ALC_FLR.3

This component is added in order to provide a higher assurance on the flow remediation. It ensures that all security flows in each release of the TOE are tracked. .This component adds confidence to customers that each security flow is described in terms of its nature and effects.

This confidence is required as the product is an open platform and will host sensitive application (a sensitive service) that require timely response of security flow reports and associated corrections to issuer of this sensitive service.

8 TOE Summary Specification

8.1 TOE Summary Specification

SF_Card_Content_Management

- loading (Section 9.3.5 of [GP22]): 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 [9]): 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 [9]: 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 [9]: 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 [9]: 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.

SF_ Confidential_Personalization

This function allows to confidentially personalize an initial set for Secure Channel Keys for a Security Domain using the services of the CASD the 2 models, Pull and Push.

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 [GP22] 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 [GP22] 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_Encryption_and_Decryption

This TSF provides the applet instances with a mechanism 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. 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, RSA, 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.

SF_Ciphered_Load_File

An Application Provider may require that its Load File to be loaded and associated to its Security Domain shall be ciphered. The Ciphered Load File Data Block privilege assigned to the Application Provider's Security Domain provides this service on behalf of the Application Provider. The key and algorithm to be used for the Load File decryption are implicitly known by the corresponding Application Provider's Security Domain.

SF_Atomic_Transactions

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 Java Card API.

This TSF also performs the actions necessary to roll back to a safe state upon interruption of the following procedures, for example because of a card withdrawal or an unexpected fatal error:

- 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.

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 [7] and in the documentation associated to the JCSystem class in the [6].

SF_Key_Management

This function enables key sets management. 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_Token

This function ensures the validity of a card content management command when it is processed by a Delegated Management Security Domain.

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_Random_Number_Generation

This TSF provides to card manager, resident 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_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_Receipt

This function provides a proof that a card content management command has been processed by a Delegated Management Security Domain.

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 [7]. 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_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 or the Remote Access Control function.

SF_Signature

This TSF provides the applet instances with a mechanism for generating an electronic signature of the contents of a byte array 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 ciphering algorithms are available to the applets through the javacard.Signature class of the Java Card API. 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, RSA, 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.

SF_Message_Digest

This TSF provides the applet instances with a mechanism for generating an (almost) unique value for the contents of a byte array. 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. Message digest generation is implemented to resist to environmental stress and glitches and include measures for preventing information leakage through covert channels.

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_Data_Integrity

Some of the data in non volatile memory can be protected. Keys, PIN package and patch code are protected with integrity value. When reading and writing operation are, the integrity value is checked and maintained valid. In case of incoherency, an exception is raise to prevent the bad use of the data.

SF_Clearing_of_Sensitive_Information

This TSF clears all the data containers that hold sensitive information when that information is no longer used. 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.
- All the information contained in the packages that is not necessary for executing the code of the applets, like the Descriptor Component, the Reference Location Component and the Constant Pool of the CAP files.
- The contents of the APDU buffer after processing an APDU command.
- The content of the bArray argument of the Applet.install method after a new applet instance is installed.
- The content of CLEAR ON DESELCT 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 reason code contained in the instances of a CardException or CardRuntimeException classes after a card reset.
- The validated flag of the PINs after a card reset.
- The contents of the cryptographic buffer after performing cryptographic operations.
- The content of the input parameters of a remote method invocation after returning the response to the terminal.

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_Key_Generation

This TSF enforces the creation and the oncard generation of all the cryptographic keys of the card using the method specified in that SFR.

SF_Key_Distribution

This TSF enforces the distribution of all the cryptographic keys of the card using the method specified in that SFR.

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, RSA, RNG implements a known 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.

SF_Memory_failure

When using the non volatile memory, in case of a bad writing, internal mechanisms are implemented to prevent an incoherency of the written data. In case of an impossible writing, an exception is raised.

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_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_Security_Functions_of_the IC

The TOE uses the security functions of the IC. The list of the security function is presented in the IC Security Target [24].

SF_Key_Access

This TSF enforces secure access to all cryptographic keys of the card: RSA keys, DES keys, AES keys.

SF_Key_Agreement

This TSF provides the applet instances with a mechanism for supporting key agreement algorithms such as Diffie-Hellman.

SF_Ciphering

This TSF provides the applet instances with a mechanism 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. 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, RSA, 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.

SF_Remote_Access

This TSF enforces the access control to remote objects when the RMI service is used. The Remote objects and its security attributes are created and initialized at the creation of the object.

SF_DF_Access_Control_OT_DSF

This SF provides an access control mechanism to the objects and security attributes that are part of the access control policy required for DESFire Access controls

SF_DF_AUT

This SF provides an authentication mechanism to separate authorized subjects from unauthorized subjects.

The authentication is performed by cryptographic algorithms.

SF_DF_CONFIDENTIALITY

This Security function provides a mechanism to protect communication against eavesdropping by encrypting communications. The encryption is required by the file owner, in the file attributes.

This function adds data to the communication stream that enables the terminal to detect integrity violations, replay attacks or man_in_the_middle attacks.

This function checks also the data send by the terminal and returns an error code if such an attack is detected.

The detection mechanism covers all frames echanged between the terminal and the card up to the current encryption frame.

SF_DF_TYPE_CONSISTANCY

This function ensures the type consistency of file types stored by the DESFire. It ensures that values cannot over-or underflow.

SF_DF_TRANSACTION

The Transaction mechanism ensures that either all or none of the commands within a transaction are performed.

8.2 SFRs and TSS

This part is removed in this public ST.

9 Compatibility

This part is removed in this public ST.