Security Target

DESFIRE 1.1 on Upteq NFC2.1.3_Generic

Table of Contents

Table of Figures

Table of Tables

1 ST Introduction

1.1 ST Reference

Security Target and associated evaluation are completely defined by information located in the following table.

Title:	Security Target : DESFIRE 1.1 on Upteq NFC2.1.3_Generic
Reference:	D1314435
Version	1.0p
Origin:	GEMALTO
ITSEF:	THALES
Certification Body:	ANSSI
Evaluation scheme:	French

Table 1 ST References

This Security Target describes:

- The Target of Evaluation, the TOE components, the components in the TOE environment, the product type, the TOE environment and life cycle, the limits of the TOE,
- The assets to be protected, the threats to be countered by the TOE itself during the usage of the TOE,
- The organizational security policies, and the assumptions,
- The security objectives for the TOE and its environment,
- The security functional requirements for the TOE and its IT environment,
- The TOE security assurance requirements,
- The security functions and associated rationales.

The assurance level for the TOE is **EAL4 augmented by AVA_VAN.5 and ALC_DVS.2.**
The minimum strength level for the TOE security functions is high (**SOF high**).

1.2 TOE Reference

Product and TOE are completely defined by information located in the following table.

Product Name	Upteq NFC 2.1.3_Generic
Product Reference	T1020806
Product Version	Release C
TOE name	DESFIRE 1.1 on Upteq NFC 2.1.3_Generic platform using ST33F1M
TOE Reference	S1124940
TOE Version	Release C
Commercial Name	UpTeq NFC2.1.3 Desfire CC

Table 2 TOE References

This product is developed as a new version of an already certified product.

1.3 TOE overview

1.3.1 TOE Type

The product **Upteq NFC 2.1.3_Generic** is (U)SIM smart card defined to be used mainly in a mobile or a Smartphone, but can be used in any device with an interface conformant to [IS0 7816] specification. It is delivered using an ISO form factor including a plug-in form factor as defined in [TS 102 221] or 3FF form factor.

The product **Upteq NFC 2.1.3_Generic** (also named m-NFC 2.1 in this document) implements the standard communication protocol (ISO 7816 T=0) and ETSI standard allowing communication between smartcard, mobile and server using OTA.

Moreover, the **Upteq NFC 2.1.3_Generic** implements the SWP [TS102613], a new full-duplex, high-speed transport protocol between the smart card and an interface device. Inserted in a NFC-enabled mobile phone, m-NFC 2.1 allows communication with a terminal using the standard ISO/IEC 14443 communication protocol.

When loaded on **Upteq NFC 2.1.3_Generic**, payment or access control or transport or loyalty applications using SWP and NFC interfaces offer convergence between Mobile communication environment with OTA administration and convenience and security of secure contact less transaction based on smartcard.

Thus, a mobile NFC payment transaction is achieved by swiping the mobile over a NFC reader at a point of sale, creating a secure connection between reader and **Upteq NFC 2.1.3_Generic** where banking application secures the transaction.

In the same manner, an access to public transport is granted to a user by swiping the mobile over a NFC reader, creating a secure connection between reader and **Upteq NFC 2.1.3_Generic** where transport application manages the access control operation.

DESFire EV1 software provides:

- A set of functions used to manage the various kinds of data files stored in the non-volatile memory, and data in RAM,
- A set of security mechanisms implemented to ensure proper operation as well as integrity and confidentiality of stored data,
- A set of functions used to establish a contactless communication or a contact communication using I/O port.

And more precisely, it provides the following functionalities:

- Flexible file system that can contain up to 28 applications with up to 32 files in each application.
- Support for different file types like values or data records.
- Mutual three pass authentication, also according to ISO 7816-4.
- Authentication on application level with fine-grained access conditions for files.
- Multi-application support that allows distributed management of applications and ensures application segregation.
- Data encryption for contact-less communication with replay attack protection.
- Transaction system with rollback that ensures consistency for complex transactions.
- Unique serial number for each device (UID) with optional random UID.

The Target of Evaluation (TOE) is the (U)SIM Java Card platform embedded in a (U)SIM card intended to be plugged in a mobile phone or other mobile devices to provide services to an end user. TOE type consistency is given in Conformance rationale in §2.3.

The Basic TOE is composed of the following bricks:

- A Java Card System according to [PP-JCS] which manages and executes applications called applets. It also provides APIs [JCAPI301] to develop applets on top of it, in accordance with Java Card ™ specifications,
- 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 [GP] specifications,
- Platform APIs, which provides ways to specifically interact with (U)SIM applications, according to [TS131.130] specifications,
- Telecom environment including network authentication applications (not evaluated) and Telecom communication protocol.
- GemActivate application to activate services in Post-Issuance (loaded in Pre-Issuance) under issuer and Gemalto administration.

The TOE configuration is defined to answer to DESFIRE requirements defined by NXP.
The TOE configuration is conformant to [PP-JCS].

1.3.2 TOE usage

Figure 1: Upteq NFC 2.1.3_Generic to be inserted in a mobile

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. A Subscriber Identity Module (SIM) is a removable module to plug within GSM mobile equipment that contains the International Mobile Subscriber Identity (IMSI) which unambiguously identifies a subscriber. It also stores other subscriber-related information or applications such as SIM Toolkit, and other application (as an E-sign application). In the rest of the document, the term of (U)SIM is used to refer to SIM or USIM as there are considered in the same way regarding security.

The primary services of the (U)SIM (when it is plugged in handset) are the user authentication by PIN capture and the SIM authentication on the MNO network, giving access to MNO services through the mobile. It also stores other subscriber-related information or applications such as SIM Toolkit applications as specified in [TS102.223] and [TS131.111].

The **Upteq NFC 2.1.3_Generic** implements major industry standards:

- Java Card 3.0.1,
- Global Platform 2.2.1 with UICC configuration 1.0.1,
- Full ETSI release 6,
- 3GPP Release 6.

It supports **multiple networks (2G, 3G,…)** and it implies that several Network Access Applications (NAA) working together, requiring for dynamic switching from networks (3G to 2G, 2G to 3G). Each application is designed like a plug-in.

The DESFire EV1 Software supports DESFire compatible applications in the field of:

- Electronic fare collection
- Stored value card systems
- Access control systems
- Loyalty

If privacy is an issue, the DESFire EV1 software can be configured not to disclose any information to unauthorized users. However in this case also the application(s) implemented in the Security IC Embedded Software must support this privacy issue. Otherwise the privacy enforced by the DESFire EV1 Software can be circumvented by selecting another application of the TOE.

A dedicated set of applets can be installed in pre-issuance phase on MNO request.

1.3.3 TOE Boundaries

The following figures illustrate the TOE physical and logical boundaries.

The product is a smartcard including a plastic card and a module performing the interface between reader and the mobile and the embedded chip. The Target of Evaluation (TOE in brown in figure TOE logical boundaries) is the Smart Card Integrated Circuit with Embedded Software in operation and in accordance to its functional specifications. Other smart card product items (such as plastic, module, bounding, printing…) are outside the scope of this evaluation.

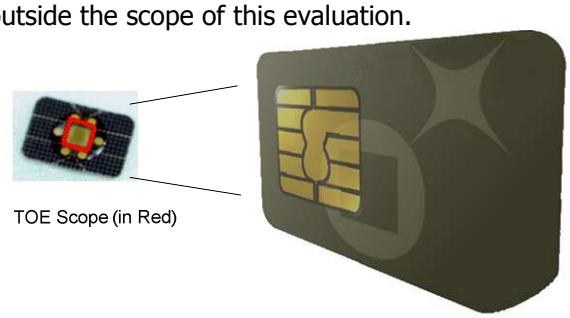

Figure 2: TOE Physical Boundaries

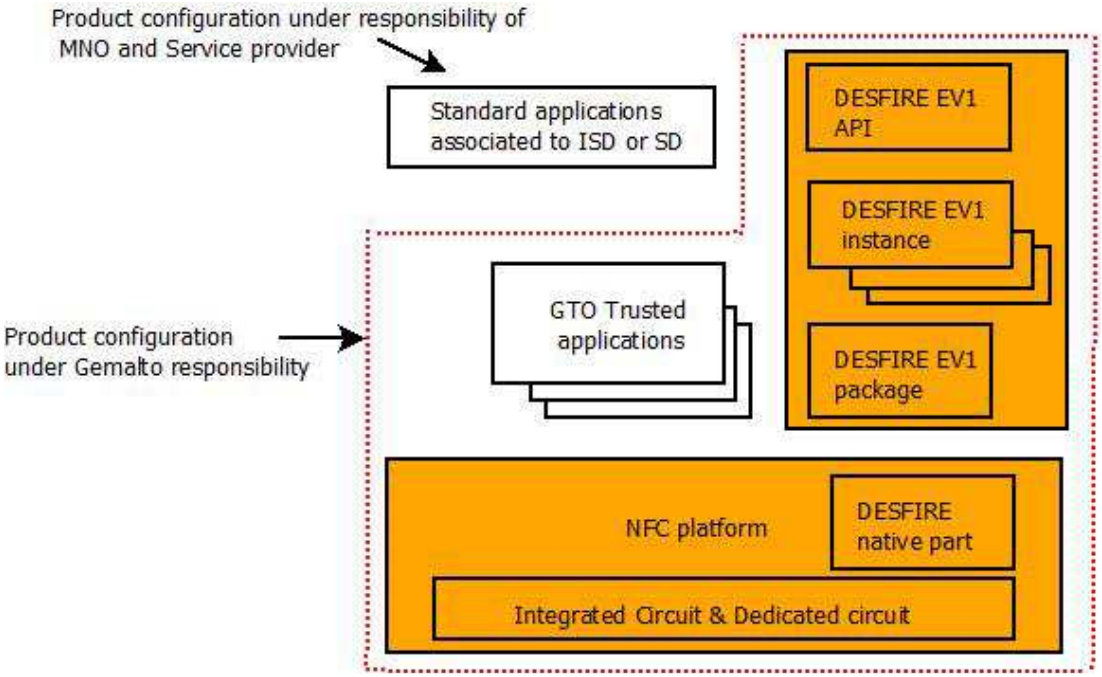

Figure 3: TOE Logical Boundaries

The NFC platform is able:

- to provide services to applications thanks to dedicated APIs (JavaCard, GP, UICC,…)
- to create and manage security domains where applications are associated
- **to instantiate and manage one or several DESFIRE applications,**
- to load and manage one or several Trusted applications developed by Gemalto,
- to load and manage one or several applications under MNO or service provider responsibility,
- to activate application by OTA.

DESFIRE application can be personalized in PRE-ISSUANCE or in POST-ISSUANCE.

1.3.4 TOE Description

The TOE contains the following components:

Component	Reference / Version	Supplier
DESFIRE 1.1 on Upteq NFC2.1.3_Generic Platform	S1124940 / Release C	Gemalto
Micro-controller ST33F1M and part of its dedicated software (DS)	Rev E	STMicroelectronics

Table 3 TOE components

The IC (Micro-controller ST33F1M) included in the TOE is compliant with the [PP-BSI-0035]. It includes a crypto library and a Flash loader used for software loading but inactivated for end user usage.

The TOE is compliant with the version of the Java card TM platform specified in [JCVM301], [JCRE301] and [JCAPI301]. It includes the Java card TM Virtual Machine (JCVM), the Java card TM Runtime Environment (JCRE) and the Java card TM Application Programming Interface (JCAPI). This set is also named Java Card System.
As the product is an open platform, the isolation mechanism between applications loaded on the TOE will be studied.

The TOE is compliant with platform APIs, which provides ways to interact with (U)SIM, SIM, UICC applications, according to [TS131.130] specifications.

The TOE provides thanks to UICC API [TS102.241] 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 TOE provides the (U)SIM API [TS131.130] extending 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.

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 enhance the SMS technology as a data bearer for mobile phones. It is specified in 3GPP specifications as [TS102.223], [TS102.225] and [TS131.111]. Note that, as specified in [PP-USIM], the BIP technology does not offer any security function for the TOE.

The TOE is compliant with the Global Platform™ standard [GP22] which provides a set of APIs and technologies to perform in secure way, the operations involved in the management of the security domains and applications hosted by 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. In addition, the TOE provides with a GP

loader applications downloading and installing operation to an end user (which must nevertheless be authenticated).

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

- Card content loading
- Extradition
- Asymmetric keys
- DAP support
- DAP calculation with asymmetric cryptography
- Logical channels
- SCP02 support
- SCP80 support defined by the ETSI [TS102.225] (mandatory for the ISD)
- Support of contactless services (ATQ, different implicit selection on different interfaces and channels)
- Support of extra Security Domains
- Installation of Security Domains
- Trusted Path privilege
- Delegated Management privilege
- Post-issuance personalization of Security Domain [GP-UICC]
- Application personalization [GP-UICC].

Note: GemActivate application is associated by default with ISD. GASD is optional and created only on MNO decision. In such case, GemActivate application can be extradited on GASD to use dedicated SCP80 secure channel. For ETSI secure scripting according to GP UICC config, by default ISD SCP80 is used, otherwise GASD or ascendant SD of GASD (e.g ISD) SCP80 is used. In both cases, GemActivate application performs applicative checks prior any required operation.

Note: The Authorized Management privilege is only supported for ISD as excluded in [PPUSIM].

The TOE includes the Telecom Environment with Network Authentication Application, Over The Air and BIP communication, File System management, and Toolkit services.

The TOE supplies secure API in native language to provide enhanced security to applets.

The TOE provides DESFIRE application and DESFIRE API to personalize DESFIRE application.

The evaluation scope is defined to demonstrate the conformity and robustness of DESFIRE EV1 implementation to the specification and access control of platform to avoid unauthorized access to DESFIRE services and data.

The following figure describes the major items included in the TOE.

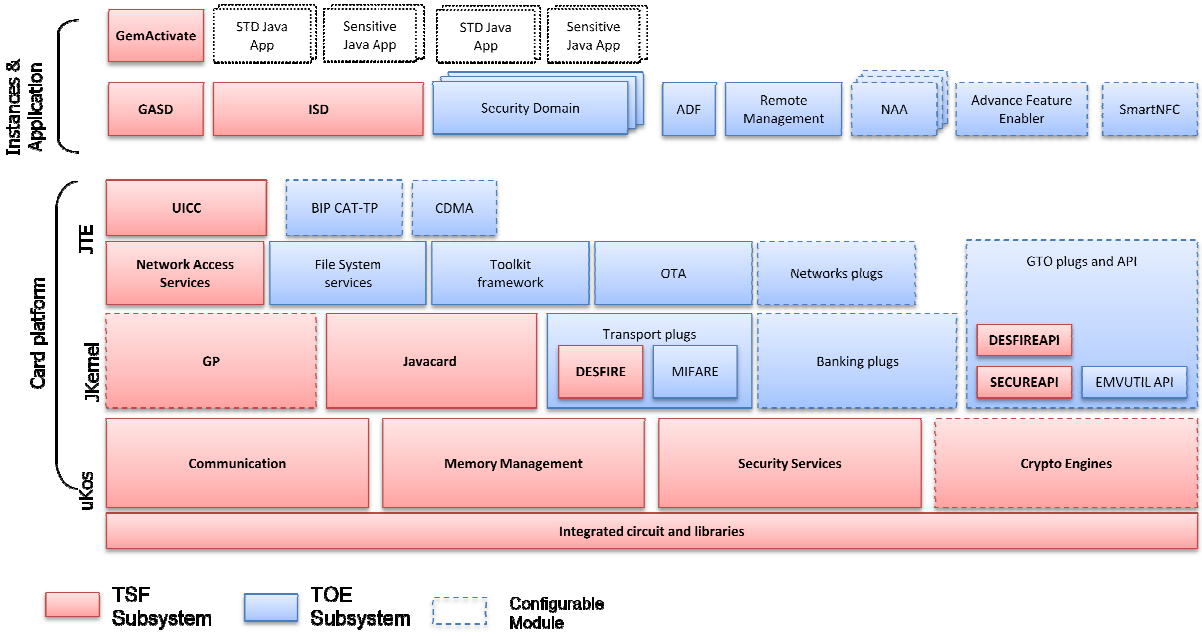

Figure 4: Major TOE Items and TOE scope

1.3.5 TOE Life Cycle

Product life cycle is described in the following picture using [PP-USIM] description refined with Gemalto specific environment due to embedded software loading in flash in phase 6.

The (U)SIM platform life cycle is composed of four stages (as defined in PP (U)SIM figure 3):

- Development (embedded software and IC separately),
- Storage, pre-personalization and test,
- Loading, Personalization and test,
- Final usage.

Refined life cycle based on [PP-BSI-0035] with Gemalto product constraints is described in the following figure.

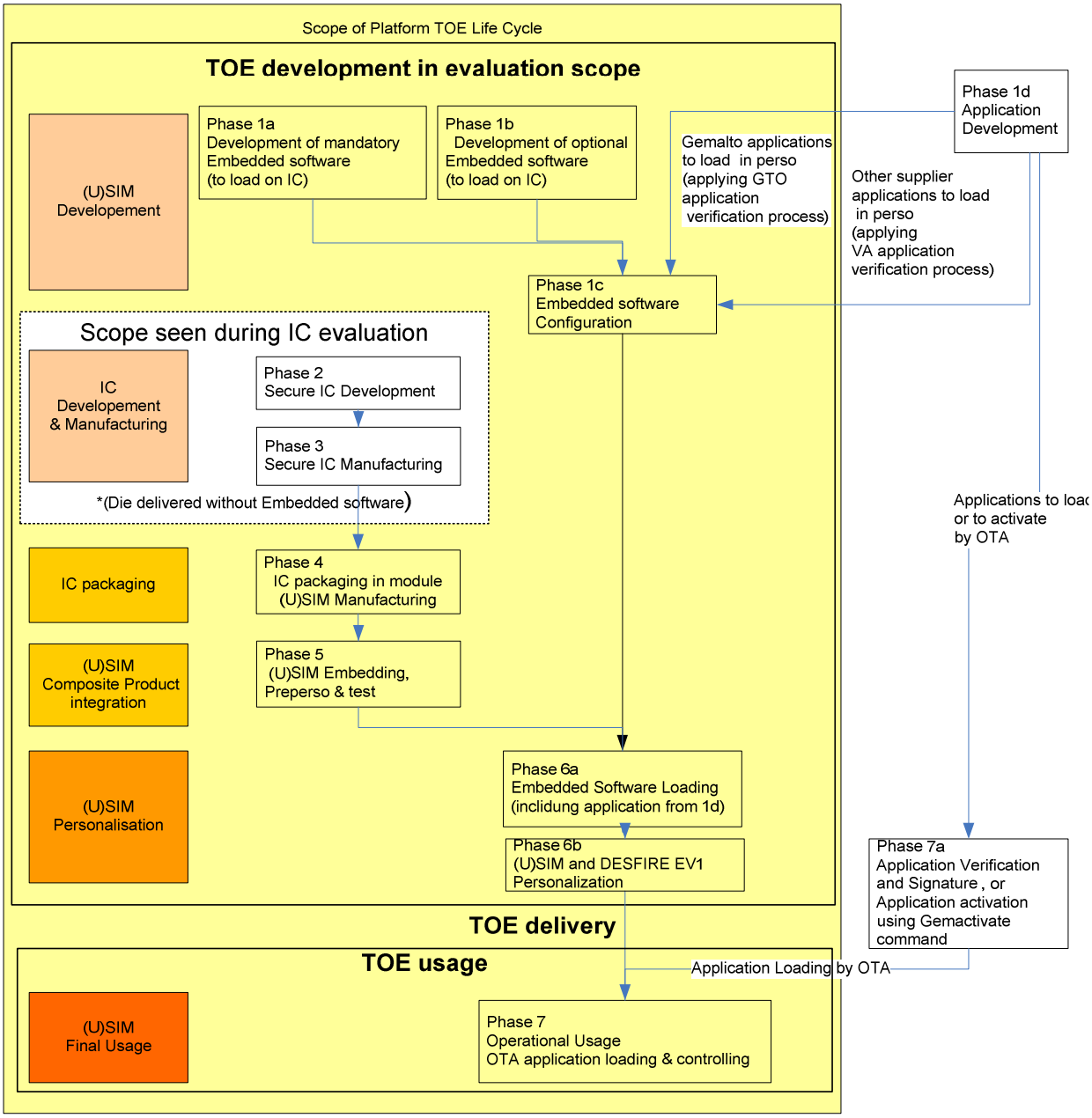

Figure 5: Refined TOE Life Cycle

The following phases corresponding to the one previously described are:

- Phases 1(a, b) correspond to the development of the TOE embedded software and its configuration (1c) with applications to be loaded in phase 6.
- Phase 2, 3 and 4 correspond to IC development, manufacturing and packaging in module, respectively.
- Phase 5 concerns the composite product integration with the module and other smart card items,
- Phase 6 (a, b) is dedicated to the TOE embedded software loading and product personalization (including DESFIRE EV1) prior to TOE delivery.
- Phase 7 is the product operational phase including application loading and controlling by Verification authority and GemActivate Administrator in specific cases.

Note: The Gemalto application will be verified using evaluated Gemalto verification process prior to be loaded in Pre-Issuance by Gemalto. In the same way, but to protect supplier intellectual property, the application provided by third party supplier must be verified and signed by verification authority* prior to be loaded in Pre-Issuance by Gemalto. Gemalto will check application signature prior to load this application in Pre-Issuance.

Note: Here, Verification authority is a role may be assumed by the Mobile Network Operator (MNO) or its representative. Verification authority assumes that verification has been done before loading by MNO or prior to deliver a token in case of delegated management.

Note: The IC used in the current life cycle does not contain any embedded software prior to phase 6. It is under protection of software security function of IC dedicated software. As generic product, the ICs are stored in personalization environment but there are not dedicated to the TOE. After loading in phase 6, IC loading service is locked and no more available after phase 6. (ref to FMT_LIM from ST_IC).

The TOE is delivered at the end of phase 6 as shown in previous figure. It is the operational Upteq NFC 2.1.3_Generic product, as a personalized smart card.

As far as the EAL4+ evaluation scope is concerned, phases 1 to 6 are considered as development and manufacturing phases of the product but the TOE is the result of these phases that can consequently be seen as phases of the TOE generation.

The TOE delivery is performed at end of phase 6 and phase 7 is the operational phase of the TOE.

Out of the TOE evaluation scope, there are also the following operations linked to the TOE:

- in phase 1(d), the application development,
- in phase 7(a), the application verification and signature by verification authority prior to application loading and the application activation using GemActivate commands.

1.3.6 TOE Environment

Considering the TOE, the environment is defined as follows:

- Development environment corresponding to phases 1 and 2;
- Production and Personalization environments corresponding to phases 3 to 6:
- Manufacturing environment including the IC test operations, IC packaging, testing and pre-personalization (phases 3 to 5),
- Personalization environment corresponding to the loading by the IC loader of the OS in the flash memory, personalization and testing of the Smart Card with the user data (phase 6).
- User environment corresponding to the card use by a subscriber on a 2G or 3G network (phase 7).

1.3.6.1 TOE Development Environment & Roles

The TOE described in this ST is developed in different places under the control of a defined administrator as indicated below:

Phase	Administrator and Location
IC design and Dedicated Software development	STMicroelectronics Sites are defined in [ST/IC]
Embedded software Development	Gemalto (Meudon, La Ciotat, Singapore)
Embedded software Configuration	Gemalto (Gemenos, Singapore, Tczew)

1.3.6.2 TOE Manufacturing Environment

The TOE described in this ST is produced in different places under the control of a defined administrator as indicated below:

Phase	Administrator and Location
IC manufacturing and Testing	STMicroelectronics Sites are defined in [ST/IC]
IC packaging	Gemalto (Pont Audemer,Singapore)
Composite Product integration	Gemalto (Pont Audemer, Tczew)

1.3.6.3 TOE Personalization Environment

The TOE described in this ST is personalized in different places under the control of a defined administrator as indicated below:

Phase	Administrator and Location
Personalization	Gemalto (Pont Audemer, Tczew)
Delivery to Final user (MNO)	From Personalization site to MNO site

1.3.6.4 TOE User Environment

Smart Cards are used in a wide range of applications to assure authorized conditional access. This specific product is to be used on terminals such as GSM and UMTS handsets or smart card readers.
The end-user environment therefore covers an unprotected environment, thus making it difficult to avoid any abuse of the TOE. The product is prepared accordingly to mitigate such attacks in this environment.
The TOE is nevertheless under the control of the MNO administration using the OTA channel. The TOE can be blocked by GP administrative commands under administrator control.

1.3.7 Actors of the TOE

One of the characteristics of the (U)SIM Java Card platforms is that several entities are represented inside 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, could manage the loading, instantiation or deletion of applications.

- The Application Provider (AP), entity or institution responsible for the applications and their associated services. It is a financial institution (a bank), a transport operator or a third party operator.
- The Controlling Authority (CA), optional 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 [GP-UICC]).
- The GemActivate Administrator (usually Gemalto), represented on the (U)SIM card, by GemActivate application and associated keys, is responsible for the optional platform service activation in Post Issuance using OTA communication channel.

1.3.8 TOE Security Features

The TOE provides **DESFIRE EV1** application but it can also manage secure or standard applets. These 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. Other administrative operations can also be done using 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 other standard applications.

1.3.8.1 Security services of Desfire EV1 software

The TOE provides an access control mechanism to the objects and security attributes of the DESFire EV1 software. The access control mechanism assigns subjects to different groups of operations on file depending on file type.

The TOE provides an authentication mechanism to separate authorized subjects from unauthorized subjects. Different roles (Administrator, Application Manager, and Application User) are associated during the authentication request by the knowledge of the respective cryptographic key.

The TOE provides integrity and confidentiality on communication and protection of execution of operation through a transaction mechanism.

The TOE also provides an access control mechanism to the objects and security attributes of the DESFire EV1 software to registered applications through a specific DESFire API.

1.3.8.2 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, 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, 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 standard 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.

1.3.8.3 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 MNO or its representative (e.g a validation laboratory) for the standard applications, or by an ITSEF for the secure applications. All loaded applications are associated at load time to an attribute allowing the verification of application integrity and authenticity by the on-card representative of the MNO 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 keysets 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 keysets are not known by the Card issuer.

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

The TOE control importation rules when loading a new package requiring verification of a DAP computed using GemActivate key if new package references one or several restricted packages.

The TOE offers activation by OTA of optional services using GemActivate. Such activation is under control of GemActivate Administrator and secure channel operation is under control of MNO.

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

The non TOE HW/SW/FW are those defined in [PP-JCS] and [PP-USIM] as:

Byte Code verifier and Verification tool available to Verification authority, mobile handset, Terminal in point of sale, Remote server for administration and Trusted network and IT system for communication.

2 Conformance claims

2.1 CC conformance claims

This Security Target has been written using CC version V3.1 release 4.
This Security Target is CC part 2 extended with the FCS_RND.1 family. All the other security requirements have been drawn from the catalogue of requirements in Part 2 [CC-2].

This Security Target is conformant with CC part 3 [CC-3].

The evaluation is performed according [CEM] and supporting documents [JIL].

The assurance requirement of this security target is **EAL4 augmented**.
Augmentation results from compliance to [PP-USIM] are the selection of:

- **ALC_DVS.2** Sufficiency of security measures,
- **AVA_VAN.5** Advanced methodical vulnerability analysis.

This is a composite evaluation, which relies on the ST33F1ME chip certificates [IC_CERTIF] and evaluation results:

- Certification done under the ANSSI scheme
- Security Targets [ST/IC] strictly conformant to IC Protection Profile [BSI-PP-2007-0035]
- Common criteria version: 3.1
- Assurance level: EAL5 augmented by ALC_DVS.2 and AVA_VAN.5.

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

2.2 PP conformance claims

This security target claims a demonstrable conformance to [PP-JCS].

This Security Target describes the composite product including an underlying IC already certified according to [ST/IC]. The ST33F1M Security Target [ST/IC] claims strict conformance to the Security IC Platform Protection Profile [BSI-PP-2007-0035], as required by this Protection Profile. The TOE includes an Integrated Circuit certified with CC EAL5 augmented with ALC_DVS.2 and AVA_VAN.5.

Refinements of [BSI-PP-2007-0035] are described in [ST/IC] and are not repeated here.

The associated evaluation is performed using composite evaluation methodology. Therefore IC relevant information is not repeated in current ST.

2.3 Conformance rationale

This paragraph presents the consistency between the security target and the Java Card System Open configuration profile Protection Profile [PP-JCS].

The TOE type consistency is assumed as TOE is a (U)SIM card conformant to referenced Javacard, GP and ETSI standards.

In addition to all elements present in [PP-JCS], this security target includes also most of the items defined in [PP-USIM] and provides DESFIRE EV1 security features defined for the TOE.

2.3.1 SPD Statement Consistency

2.3.1.1 Assets

All assets from the protection profile [PP-JCS] are included in the security target. The following table shows Assets in addition to the Assets found in PP [PP-JCS].

(U)SIM	
D.ISD_KEYS	[PP-USIM]
D.CARD_MNGT_DATA	[PP-USIM]
D.OPTIONAL_PF_SERVICE	is added to allow service configuration
D.GASD_KEYS	Gemalto Security Domain cryptographic keys needed to authorize activation requests
DESFIRE EV1 Software	
D.DESFire_KEYS	Additional Assets regarding the DESFIRE Application
D.DESFire_FILES	
D.DESFire_VALUES	
D.DESFire_SOFTWARE	

2.3.1.2 Threats

All threats from the protection profile [PP-JCS] are included in the security target. The following table shows Threats in addition to the Threats found in PP [PP-JCS].

(U)SIM	
T.INTEG-USER-DATA	[PP-USIM]
T.UNAUTHORIZED_CARD_MNGT	[PP-USIM]
T.LIFE_CYCLE	[PP-USIM]
T.UNAUTHORIZED_ACCESS_TO_SERVICE	is associated to the asset D.OPTIONAL_PF_SERVICE
DESFire EV1 Software	
T. DESFire_DATA-MODIFICATION	Additional Threats regarding the DESFIRE Application
T. DESFire_IMPERSONATE	
T. DESFire_CLONING	
T. DESFire_API_CONFID_APPLI-Data2	
T. DESFire_API_INTEG_APPLI-Data2	
T. DESFire_API_INSTALL	

2.3.1.3 OSPs

All OSPs from the protection profile [PP-JCS] are included in the security target. The following table shows OSPs in addition to the OSPs found in PP [PP-JCS].

(U)SIM	
OSP.BASIC-APPS-VALIDATION	[PP-USIM]
OSP.AID-MANAGEMENT	[PP-USIM]
OSP.OTA-LOADING	[PP-USIM]
OSP.OTA-SERVERS	[PP-USIM]
OSP.OPERATOR-KEYS	[PP-USIM]
OSP.KEY-GENERATION	[PP-USIM]
OSP.PRODUCTION	[PP-USIM]
OSP.PERSONALIZER	[PP-USIM]
OSP.KEY-ESCROW	[PP-USIM]
OSP.SecureAPI	There are extra OSP (OSP.RNG, OSP.JCAPI-Services) to provide additional services to applications. The extra OSP (OSP.TRUSTED-APPS-DEVELOPER, OSP.TRUSTED-APPS-PRE-ISSUANCE LOADING) are provided to manage pre-issuance and the OSP (OSP.SERVICE AUDIT, OSP.ACTIVATION-KEY-ESCROW) to manage service activation by OTA. Such extension has no impact on PP coverage.
OSP.RNG	
OSP.JCAPI-Services	
OSP.TRUSTED-APPS-DEVELOPER	
OSP.TRUSTED-APPS-PRE-ISSUANCE LOADING	
OSP.SERVICE_AUDIT	
OSP.ACTIVATION-KEY-ESCROW	
DESFire EV1 Software	
OSP.DESFire_EMULATION	Additional OSPs regarding the DESFIRE Application
OSP.DESFire_Add-HW_Components	
OSP.DESFire_Add-MemorySeparation	
OSP.DESFire_KEY-FUNCTION Usage of Key-dependent Functions	
OSP.DESFire_SECURE-VALUES Usage of secure values	
OSP.DESFire_API_DESFire_APPLICATION-REGISTRATION	
OSP.DESFire_API_APPLICATION_GEMACTIVATE_DAP_GENERATION	

2.3.1.4 Assumptions

All the assumptions from the protection profile [PP-JCS] have been added in the security target, except A.DELETION. The following table shows As in addition to the Assumptions found in PP [PP-JCS].

(U)SIM	
A.MOBILE-OPERATOR	[PP-USIM]
A.OTA-ADMIN	[PP-USIM]
DESFire EV1 Software	
A.DESFire_TERMINAL-SUPPORT Terminal support to ensure integrity and confidentiality	Additional Assumptions regarding the DESFIRE Application
A.DESFire_Plat-Appl Usage of Hardware Platform	
A.DESFire Resp-Appl Treatment of User Data	

The scope of the Embedded Software addressed in this Security Target has been enlarged with respect to [PP-JCS] so as to include the Card Manager. As a consequence, the A.DELETION assumption is not applicable to the environment since it is upheld by the objective O.CARD-MANAGEMENT which controls the access to card management functions such as deletion of applets. This security objective for the current TOE corresponds to security objective for the environment OE.CARD-MANAGEMENT from the Java Card Protection Profile [PP-JCS]. As Java Card specifications do not address the deletion of Executable Files or applet instances, the A.DELETION assumption assumes that this procedure is performed safely. In this Security Target, that assumption is discharged by the threat T.DELETION. The security problem definition of the current TOE is more restrictive than the security problem definition of the Protection profile [PP-JCS].

2.3.2 Objective Statement Consistency

The security objectives statement consistency is assumed because TOE objectives and objectives for environment are aligned between the ST and the PP [PP-JCS].

2.3.2.1 Objectives for the TOE

All Os from the protection profile [PP-JCS] are included in the security target. The following table shows Os in addition to the Os found in PP [PP-JCS].

[PP-JCS]	
O.CARD-MANAGEMENT	This security objective comes from a security objective for the operational environment OE.CARD-MANAGEMENT
O.SCP.SUPPORT	This security objective comes from a security objective for the operational environment OE.SCP.SUPPORT
O.SCP.RECOVERY	This security objective comes from a security objective for the

	operational environment OE.SCP.RECOVERY
O.SCP.IC	This security objective comes from a security objective for the operational environment OE.SCP.IC
(U)SIM	
O.APPLI-AUTH	[PP-USIM]
O.COMM_AUTH	[PP-USIM]
O.COMM_INTEGRITY	[PP-USIM]
O.Secure_API	to provide additional services to applications. Such extension has no impact on PP coverage.
O.RND	
O.JCAPI-Services	
O.REMOTE_SERVICE_AUDIT	
O.REMOTE_SERVICE_ACTIVATION	
DESFire EV1 Software	
O.DESFire_MF-FW DESFIRE Firewall	Additional Os regarding the DESFIRE Application
O.DESFire_MEM-ACCESS Area based Memory Access Control	
O.DESFire_DATA-ACCESS Access Control to DESFire Data	
O.DESFire_AUTHENTICATION Authentication	
O.DESFire_CONFIDENTIALITY Confidential Communication	
O.DESFire_TYPE-CONSISTENCY Data type consistency	
O.DESFire_TRANSACTION Transaction mechanism	
O.DESFire_TDES Triple DES Functionality	
O.DESFire_AES AES Functionality	
O.DESFire_SCP_Support	
O.DESFire_API_GEMACTIVATE_DAP_VERIFICATION	
O.DESFire_API_RESTRICTED ACCESS	

2.3.2.2 Objectives for the Operational Environment

All OEs from the protection profile [PP-JCS] are included in the security target. The following table shows OEs in addition to the OEs found in PP [PP-JCS].

(U)SIM	
OE.MOBILE-OPERATOR	[PP-USIM]
OE.OTA-ADMIN	[PP-USIM]
OE.KEY-ESCROW	[PP-USIM]
OE.PERSONALIZER	[PP-USIM]
OE.GEMACTIVATE-ADMIN	[PP-USIM]
OE.PRODUCTION	[PP-USIM]
OE.BASIC-APPS-VALIDATION	[PP-USIM]

OE.AID-MANAGEMENT	[PP-USIM]
OE.OTA-LOADING	[PP-USIM]
OE.OTA-SERVERS	[PP-USIM]
OE.OPERATOR-KEYS	[PP-USIM]
OE.KEY-GENERATION	[PP-USIM]
OE.TRUSTED-APPS-DEVELOPER	are added to manage application development, pre-issuance loading and service activation covering the associated organizational security policies.
OE.TRUSTED-APPS-PRE-ISSUANCE LOADING	
OE.ACTIVATION-KEY-ESCROW	
DESFire EV1 Software	
OE.DESFire_RESP-APPL Treatment of User Data	Additional OEs regarding the DESFIRE Application
OE.DESFire_Plat-APPL Usage of Hardware Platform	
OE.DESFire_SECURE-VALUES Generation of secure values	
OE.DESFire_TERMINAL-SUPPORT Terminal support to ensure integrity and confidentiality	
OE.DESFire_API_CODEC-AID-INIT	

Objectives for environment in [PP-JCS] become objectives for the TOE in ST due to inclusion of IC and OS in the ST scope. It is the case for OE.CARD-MANAGEMENT, OE.SCP.IC, OE.SCP.RECOVERY, and OE.SCP.SUPPORT becoming respectively O.CARD-MANAGEMENT, O.SCP.IC, O.SCP.RECOVERY, and O.SCP.SUPPORT.

2.3.3 SFR Statement Consistency

All SFRs from the protection profile [PP-JCS] are included in the security target. The following table shows SFRs in addition to the SFRs found in PP [PP-JCS].

SFR related to PP (U)SIM	
FDP_ITC.2/CCM	[PP-USIM]
FDP_ROL.1/CCM	[PP-USIM]
FDP_UIT.1/CCM	[PP-USIM]
FPT_FLS.1/CCM	[PP-USIM]
FDP_ACC.1/SD	[PP-USIM]
FDP_ACF.1/SD	[PP-USIM]
FMT_MSA.1/SD	[PP-USIM]
FMT_MSA.3/SD	[PP-USIM]
FMT_SMF.1/SD	[PP-USIM]
FMT_SMR.1/SD	[PP-USIM]
FCO_NRO.2/SC	[PP-USIM]
FDP_IFC.2/SC	[PP-USIM]
FDP_IFF.1/SC	[PP-USIM]

FIA_UID.1/SC	[PP-USIM]
FIA_UAU.1/SC	[PP-USIM]
FIA_UAU.4/SC	[PP-USIM]
FMT_MSA.1/SC	[PP-USIM]
FMT_MSA.3/SC	[PP-USIM]
FMT_SMF.1/SC	[PP-USIM]
FTP_ITC.1/SC	[PP-USIM]
SFR related to ST SCP Group	
FPT_RCV.3/OS	is added to ensure the return of the TOE to a secure state using automated procedures
FPT_RCV.4/OS	is added to ensure function recovery
SFR related to Crypto API Group	
FCS_COP.1/SHA2	Cryptographic operation: computation of a hash value for applet instance's data
FCS_COP.1/CRC	Cryptographic operation: computation of checksum CRC16 or CRC32 for applet instance's data
FCS_RND.1	provides a mechanism to generate random numbers
SFR related to Secure API Group	
FPT_FLS.1/SecureAPI	preserves a secure state when the application fails to perform a specific execution flow control protected by the Secure API
FPT_ITT.1/SecureAPI	protects TSF data from disclosure and modification when it is transmitted between separate parts of the TOE
FPR_UNO.1/SecureAPI	Enables unobservability of an external attacker
SFR related to GemActivate Group	
FMT_SMR.1/GemActivate	Additional SFRs regarding the GemActivate service
FMT_SMF.1/GemActivate	
FMT_MOF.1/GemActivate	
FMT_MSA.1/GemActivate	
FMT_MTD.1/GemActivate	
FDP_ACC.1/GemActivate_DAP	
FDP_ACF.1/GemActivate_DAP	
FMT_MSA.3/GemActivate_DAP	
SFR related to DESFIRE Group	
FDP_ACC.1/DESFire	Additional SFRs regarding the DESFire application
FDP_ACF.1/DESFire	
FCS_COP.1/DESFire_TDES	
FCS_COP.1/DESFire_AES	
FMT_MSA.3/DESFire	
FMT_MSA.1/DESFire	
FMT_SMF.1/DESFire	
FMT_SMR.1/DESFire	
FDP_ITC.2/DESFire	
FPT_TDC.1/DESFire	
FIA_UID.2/DESFire	

FIA_UAU.2/DESFire	
FIA_UAU.5/DESFire	
FMT_MTD.1/DESFire	
FTP_TRP.1/DESFire	
FCS_CKM.4/DESFire	
FDP_ROL.1/DESFire	
FPT_RPL.1/DESFire	
FDP_ACC.1/DESFire_API	
FDP_ACF.1/DESFire_API	
FMT_SMR.1/DESFire_API	
FMT_SMF.1/DESFire_API	
FDP_ETC.1/DESFire_API	
FDP_ITC.1/DESFire_API	
FIA_UID.2/DESFire_API	
FMT_MSA.1/DESFire_API	
FMT_MSA.3/DESFire_API	

2.3.4 SAR Statement Consistency

The SARs statements consistency is assumed because the same assurance package is used.

3 Security aspects

This chapter describes the main security issues of the Java Card System and its environment addressed in this Security Target, called "security aspects", in a CC-independent way. In addition to this, they also give a semi-formal framework to express the CC security environment and objectives of the TOE. They can be instantiated as assumptions, threats, objectives (for the TOE and the environment) or organizational security policies.

For instance, we will define hereafter the following aspect:

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

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

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

3.1 Confidentiality

#.CONFID-APPLI-DATA Application data must be protected against unauthorized disclosure. This concerns logical attacks at runtime in order to gain read access to other application's data.

#.CONFID-JCS-CODE Java Card System code must be protected against unauthorized disclosure. Knowledge of the Java Card System code may allow bypassing the TSF. This concerns logical attacks at runtime in order to gain a read access to executable code, typically by executing an application that tries to read the memory area where a piece of Java Card System code is stored.

#.CONFID-JCS-DATA Java Card System data must be protected against unauthorized disclosure. This concerns logical attacks at runtime in order to gain a read access to Java

Card System data. Java Card System data includes the data managed by the Java Card RE, the Java Card VM and the internal data of Java Card platform API classes as well.

3.2 Integrity

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

#.INTEG-APPLI-DATA Application data must be protected against unauthorized modification. This concerns logical attacks at runtime in order to gain unauthorized write access to application data. In post-issuance application loading, this threat also concerns the modification of application data contained in a package in transit to the card. For instance, a package contains the values to be used for initializing the static fields of the package.

#.INTEG-JCS-CODE Java Card System code must be protected against unauthorized modification. This concerns logical attacks at runtime in order to gain write access to executable code.

#.INTEG-JCS-DATA Java Card System data must be protected against unauthorized modification. This concerns logical attacks at runtime in order to gain write access to Java Card System data. Java Card System data includes the data managed by the Java Card RE, the Java Card VM and the internal data of Java Card API classes as well.

3.3 Unauthorized Executions

#.EXE-APPLI-CODE Application (byte)code must be protected against unauthorized execution. This concerns (1) invoking a method outside the scope of the accessibility rules provided by the access modifiers of the Java programming language ([JAVASPEC], §6.6); (2) jumping inside a method fragment or interpreting the contents of a data memory area as if it was executable code; (3) unauthorized execution of a remote method from the CAD (if the TOE provides JCRMI functionality).

#.EXE-JCS-CODE Java Card System bytecode must be protected against unauthorized execution. Java Card System bytecode includes any code of the Java Card RE or API. This concerns (1) invoking a method outside the scope of the accessibility rules provided by the access modifiers of the Java programming language ([JAVASPEC], §6.6); (2) jumping inside a method fragment or interpreting the contents of a data memory area as if it was executable code. Note that execute access to native code of the Java Card System and applications is the concern of #.NATIVE.

#.FIREWALL The Firewall shall ensure controlled sharing of class instances7, and isolation of their data and code between packages (that is, controlled execution contexts) as well as between packages and the JCRE context. An applet shall not read, write, compare a piece of

data belonging to an applet that is not in the same context, or execute one of the methods of an applet in another context without its authorization.

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

4 Security problem definition

4.1 Assets

4.1.1 (U)SIM Java card TM Platform Protection Profile

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.

The assets introduced in [PP-JCS] are of two kinds: specialisation of a PPJCS asset (all the threats identified in JCS apply, plus some new ones) or new asset (new threats apply).

They are divided first following the two configurations and then 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.

Note that assets listed in the underlying Java Card System Protection Profile are included in this Security Target. For a detailed description refer to [PP-JCS].

4.1.1.1 Basic TOE

This section describes the assets for the Basic TOE.

User Data

The following assets specialize the asset D.APP_KEYs from [PP-JCS].

D.ISD_KEYS

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

To be protected from unauthorized disclosure and modification.

TSF Data

D.CARD_MNGT_DATA

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

To be protected from unauthorized modification.

4.1.2 Java Card System Protection Profile - Open Configuration

Assets are security-relevant elements to be directly protected by the TOE. Confidentiality of assets is always intended with respect to un-trusted people or software, as various parties

are involved during the first stages of the smart card product life-cycle; details are given in threats hereafter.

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

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

4.1.2.1 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 (U)SIM

D.OPTIONAL_PF_SERVICE

Platform services can be configured by addition of optional services as:

new cryptographic algorithm service available through API

new network authentication algorithm available through API

D.GASD_KEYS

Gemalto Security Domain cryptographic keys needed to authorize activation requests.

To be protected from unauthorized disclosure and modification.

4.1.4 DESFIRE EV1

D.DESFire_KEYS

the Keys controlled by the DESFire EV1 Software.

D.DESFire_FILES

the Files controlled by the DESFire EV1 Software.

D.DESFire_VALUES

the Values controlled by the DESFire EV1 Software.

D.DESFire_SOFTWARE

the DESFire EV1 Software, stored and in operation.

4.2 Users / Subjects

4.2.1 (U)SIM Java card TM Platform Protection Profile

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 [PP-JCS] plus the following ones:

4.2.1.1 Basic TOE

This section describes the subjects for the Basic TOE, applicable to [PPUSIM].

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

S.GEMACTIVATE
GemActivate Security Domain representing on the card a Gemalto administrator. This entity can activate application in Post-Issuance and can authorize loading of application importing restricted package.

4.2.2 Java Card System Protection Profile - Open Configuration

The subjects associated to JCS are defined in chapter introduction related to JCS security functional requirements.

4.2.3 DESFIRE EV1

S.DESFIRE
DESFIRE Application. This entity is the representative of the Application Manager in the TOE.

S.DESFIRE_API
DESFIRE API. This entity allows access to DESFIRE application service to defined applications.

4.3 Threats

4.3.1 (U)SIM Java card TM Platform Protection Profile

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 threats listed below focus only on the (U)SIM Platform. Some of them refine those already present in [PP-JCS].

All the Java Card System threats of [PP-JCS] are also relevant to this Security Target.

4.3.1.1 Basic TOE

This section describes threats for the Basic TOE, applicable to [PPUSIM].

T.PHYSICAL

The attacker discloses or modifies the design of the TOE, its sensitive data or application code by physical (opposed to logical) tampering means. This threat includes IC failure analysis, electrical probing, unexpected tearing, and DP analysis. That also includes the modification of the runtime execution of Java Card System, GlobalPlatform or SCP or SCWS (for the SCWS TOE) software through alteration of the intended execution order of (set of) instructions through physical tampering techniques.

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

This threatens all the identified assets.

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

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.ISD_KEYS and D.GASD_KEYS**

T.UNAUTHORIZED_CARD_MNGT

The attacker performs unauthorized card management operations (for instance impersonates one of the actors 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.GASD_KEYS**, **D.APP_C_DATA** (from [PP-JCS]), **D.APP_I_DATA** (from [PP-JCS]), **D.APP_CODE** (from [PP-JCS]) 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** (from [PP-JCS]), **D.APP_C_DATA** (from [PP-JCS]), and **D.CARD_MNGT_DATA**.

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. Several groups of threats are distinguished according to the configuration chosen for the TOE and the means used in the attack. The classification is also inspired by the components of the TOE that are supposed to counter each threat.

4.3.2.1 CONFIDENTIALITY

T.CONFID-APPLI-DATA

The attacker executes an application to disclose data belonging to another application. See #.CONFID-APPLI-DATA for details.

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

T.CONFID-JCS-CODE

The attacker executes an application to disclose the Java Card System code. See #.CONFID-JCS-CODE for details.

Directly threatened asset(s): D.JCS_CODE.

T.CONFID-JCS-DATA

The attacker executes an application to disclose data belonging to the Java Card System. See #.CONFID-JCS-DATA for details.

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

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. See #.INTEG-APPLI-CODE for details.

Directly threatened asset(s): D.APP_CODE.

T.INTEG-APPLI-CODE.LOAD

The attacker modifies (part of) its own or another application code when an application package is transmitted to the card for installation. See #.INTEG-APPLI-CODE for details.

Directly threatened asset(s): D.APP_CODE.

T.INTEG-APPLI-DATA

The attacker executes an application to alter (part of) another application's data. See #.INTEG-APPLI-DATA for details.

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

T.INTEG-APPLI-DATA.LOAD

The attacker modifies (part of) the initialization data contained in an application package when the package is transmitted to the card for installation. See #.INTEG-APPLI-DATA for details.

Directly threatened asset(s): D.APP_I_DATA and D_APP_KEY.

T.INTEG-JCS-CODE

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

Directly threatened asset(s): D.JCS_CODE.

T.INTEG-JCS-DATA

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

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

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

4.3.2.3 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. See #.SID for details.

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

T.SID.2

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

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

4.3.2.4 UNAUTHORIZED EXECUTION

T.EXE-CODE.1

An applet performs an unauthorized execution of a method. See #.EXE-JCS-CODE and #.EXE-APPLI-CODE for details.

Directly threatened asset(s): D.APP_CODE.

T.EXE-CODE.2

An applet performs an execution of a method fragment or arbitrary data. See #.EXE-JCS-CODE and #.EXE-APPLI-CODE for details.

Directly threatened asset(s): D.APP_CODE.

T.NATIVE

An applet executes a native method to bypass a TOE Security Function such as the firewall. See #.NATIVE for details.

Directly threatened asset(s): D.JCS_DATA.

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. See #.RESOURCES for details.

Directly threatened asset(s): D.JCS_DATA.

4.3.2.6 CARD MANAGEMENT

T.DELETION

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

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

Application note:

Due to Protection Profile and ST definition, T.DELETION replaces A.DELETION as O.CARD_MANAGEMENT replaces OE.CARD_MANAGEMENT.

T.INSTALL

The attacker fraudulently installs post-issuance of an applet on the card. This concerns either the installation of an unverified applet or an attempt to induce a malfunction in the TOE through the installation process. See #.INSTALL for details.

Directly threatened asset(s): D.SEC_DATA (any other asset may be jeopardized should this attack succeed, depending on the virulence of the installed application).

4.3.2.7 SERVICES

T.OBJ-DELETION

The attacker keeps a reference to a garbage collected object in order to force the TOE to execute an unavailable method, to make it to crash, or to gain access to a memory

containing data that is now being used by another application. See #.OBJ-DELETION for further details.

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

4.3.2.8 DENIAL OF SERVICE

T.RESOURCES

An attacker prevents correct operation of the Java Card System through consumption of some resources of the card: RAM or NVRAM. See #.RESOURCES for details.

Directly threatened asset(s): D.JCS_DATA.

4.3.3 (U)SIM

T.UNAUTHORIZED_ACCESS_TO_SERVICE

An attacker may gain direct access to an optional platform service without authorization by bypassing access control to service activation. Directly threatened asset(s): D.GASD_KEYS, D.OPTIONAL_PF_SERVICE

4.3.4 DESFire EV1 Software

This security target defines additional threats related to the functionality provided by the DESFire EV1 Software.

T. DESFire_DATA-MODIFICATION

Unauthorized modification of keys, files and values maintained by the DESFire EV1 Software. Keys, files and values maintained by the DESFire EV1 Software are processed and stored by the TOE. They may be modified by unauthorized subjects. This threat applies to the processing of modified commands received by the TOE, it is not concerned with verification of authenticity.

Application note: (asset: keys, files and values maintained by the DESFire EV1,

threat agent: unauthorized subjects)

T. DESFire_IMPERSONATE

Impersonating authorized users during the authentication process of the DESFire EV1 software. An unauthorized subject may try to impersonate an authorized subject during the authentication sequence of the DESFire EV1 Software, e.g. by a man-in-the middle or replay attack.

Application note: (asset: DESFire EV1 methods, threat agent: unauthorized subjects)

T. DESFire_CLONING

Cloning using keys, files and values maintained by the DESFire EV1 Software Keys, files and values maintained by the DESFire EV1 Software stored on the TOE may be read out by an unauthorized subject in order to create a duplicate.

Application note: (asset: DESFire EV1 Software Keys, files and values, threat agent: unauthorized subjects)

T. DESFire_API_CONFID_APPLI-Data2

If the platform configuration allows the post-issuance of data using DESFIRE API, it may lead to unauthorized disclosure of DESFire EV1 data when they are transmitted to the card for installation.

Application note: (asset: DESFire EV1 data, threat agent: unauthorized users or applet)

T. DESFire_API_INTEG_APPLI-Data2

If the platform configuration allows the post-issuance of data using DESFIRE API, it may lead to unauthorized modification of DESFire EV1 data when they are transmitted to the card for installation.

Application note: (asset: DESFire EV1 data, threat agent: unauthorized users or applet)

T. DESFire_API_INSTALL

If the platform configuration allows the post-issuance of code referencing DESFIRE API, it may lead to unauthorized access to DESFire EV1 methods from fraudulent code installations. This concerns the installation of an unauthorized application referencing DESFIRE API through the installation process.

Application note: (asset: access to DESFire EV1 methods, threat agent: fraudulent code = unauthorized applet)

4.4 Organisational Security Policies

4.4.1 (U)SIM Java card TM Platform Protection Profile

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 shall comply when addressing security needs related to (U)SIM Java Card Platform.

All the OSPs listed in [PP-JCS] are relevant for this Protection Profile.

This Security Target adds the following OSPs:

4.4.1.1 Basic TOE

This section describes OSPs for the Basic TOE.

Standard and secure applications policies

This Protection Profile distinguishes standard 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. Standard and Secure applets are loaded in different Java Card packages.

OSP.BASIC-APPS-VALIDATION

Standard 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 standard applications:

must follow the extra-rules stated in the user manual of the considered (U)SIM Java Card Platform,

cannot be libraries,

must not use RMI,

must not 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,

must be associated to an identifier and this identifier has to be used in parameter of the function calls.

Application note:

GSM file system and API's STK application descriptors are other ways to share object between applications.

Identifier usage allows to easily track applications calls. This is useful if a new attack path is discovered to identify the pieces of code that could be vulnerable.

See [Standard APP] for more details on the validation process.**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.

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, the Card issuer can pre-authorize content loading operation through delegated management 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 policy enforced by the mobile operator to ensure the security of the application can use mechanisms such as access control, isolation, regular check of integrity and encryption.

One possible realisation of this Organizational Security Policy is the enforcement of security rules defined in OTA servers security guidance document with regular site inspections to check the applicability of the rules.

Key Policies

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.Secure places

OSP.PRODUCTION

Production and personalization environment has to be secured as the TOE delivery occurs after Phase 6.

Application note:

Such OSP replaces A.PRODUCTION defines in the PP (U)SIM.

OSP.PERSONALIZER

The personalizer under an Operator's Contract 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).

Application note:

Such OSP replaces A.PERSONALIZER defines in the PP (U)SIM.

OSP.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. He ensures the security of the keys.

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 generation of signing* by the verification authority. See #.VERIFICATION for details.

If the application development guidance provided by the platform developer contains recommendations related to the isolation property of the platform, this policy shall also ensure that the verification authority checks that these recommendations are applied in the application code.

Application note: signing may be alternatively associated to the generation of integrity and authenticity evidences.

4.4.3 (U)SIM

This section describes the organizational security policies to be enforced with respect to the TOE (U)SIM environment.

OSP.SecureAPI

The TOE must contribute to ensure that application can optimize control on its sensitive operations using a dedicated API provided by TOE. TOE will provide services for secure array management and to detect loss of data integrity and inconsistent execution flow and react against tearing or fault induction.

OSP.RNG

This policy shall ensure the entropy of the random numbers provided by the TOE to applet using [JCAPI301] is sufficient. Thus attacker is not able to predict or obtain information on generated numbers.

OSP.JCAPI-Services

This policy shall ensure that hashing and checksum security services defined in [JCAPI301] provided by the TOE to applet is secure. Thus attacker is not able to predict or obtain information on manipulated data.

OSP.TRUSTED-APPS-DEVELOPER

There are application developers (as Gemalto) considered as trusted by platform issuer and application providers. The confidence in these actors has been obtained by audit of development process and development environment performed by ITSEF during private scheme evaluation or Common Criteria composite evaluation process.

Application note: As a consequence, the development process applied by a trusted developer provides confidence that applications developed by such actors are considered as not aggressive versus the platform and other applications loaded on the platform.

OSP.TRUSTED-APPS-PRE-ISSUANCE LOADING

For Pre-Issuance loading of trusted* applications, the audited process during Platform evaluation must be used.

[* Application notes: An application is considered as trusted if it has been developed or verified by a trusted actor (as Gemalto). An application developed by a third party can be considered as a trusted application only it has been verified and signed by verification

authority. The application and associated signature will be verified by Gemalto prior authorizing loading in pre-issuance.

As a consequence, the loading process applied by a trusted personalizer provides confidence that applications developed by trusted actors are considered as not aggressive versus the platform and other applications loaded on the platform.]

OSP.SERVICE_AUDIT

The MNO and GemActivate administrator (usually Gemalto) can audit optional platform service activation using remote service audit.

OSP.ACTIVATION-KEY-ESCROW

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

4.4.4 DESFire EV1 Software

OSP.DESFire_EMULATION

The DESFire emulation provides the following specific security components:

Confidentiality during communication provides the possibility to protect selected data elements from eavesdropping during contactless communication.

Integrity during communication provides the possibility to protect the contactless communication from modification or injections. This includes especially the possibility to detect replay or man-in-the-middle attacks within a session.

Transaction mechanism provides 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.

OSP.DESFire_Add-HW_Components

The DESFire emulation requires additional specific security components:

Triple-DES encryption and decryption,

AES encryption and decryption.

OSP.DESFire_Add-MemorySeparation

The DESFire emulation requires additional specific security components:

Memory separation for different software parts (including IC Dedicated Software and Security IC Embedded Software).

OSP.DESFire_KEY-FUNCTION Usage of Key-dependent Functions

Key-dependent functions (if any) shall be implemented in the Security Embedded Software in a way that they are not susceptible to leakage attacks (as described under T.Leak-Inherent and T.Leak-Forced).

OSP.DESFire_SECURE-VALUES Usage of secure values

Only confidential and secure keys shall be used to set up the authentication and access rights for the DESFire EV1 Software. These values are generated outside the TOE. They

must be protected during generation, management outside the TOE and downloaded to the TOE.

OSP.DESFire_API_APPLICATION-REGISTRATION Registration using CODEC AID

This policy ensures that any package using DESFIRE API has been registered by its CODEC AID to DESFIRE EV1 prior to have access to interface of DESFIRE API.

OSP.DESFire_API_APPLICATION_GEMACTIVATE_DAP_GENERATION

This policy ensures that computation of DAP using GemActivate Key is mandatory to allow loading of package referencing restricted packages and in particular the DESFIRE API.

4.5 Assumptions

4.5.1 (U)SIM Java card TM Platform Protection Profile

The following assumption concerns the product operational environment, after product delivery.

All the assumptions mentioned in the [PP-JCS] Protection Profile are relevant.

This Security Target adds the following assumptions:

4.5.1.1 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.

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.

4.5.2 Java Card System Protection Profile - Open Configuration

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

A.APPLET

Applets loaded post-issuance do not contain native methods. The Java Card specification explicitly "does not include support for native methods" ([JCVM301], §3.3) outside the API.

A.VERIFICATION

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

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.

4.5.3 DESFire EV1 Software

A.DESFire_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.

A.DESFire_Plat-Appl Usage of Hardware Platform

This assumption is defined in [PP-IC] and required for composite evaluation.

A.DESFire Resp-Appl Treatment of User Data

This assumption is defined in [PP-BSI-0035] and it is extended to DESFIRE application.

5 Security Objectives

5.1 Security Objectives for the TOE

5.1.1 (U)SIM Java card TM Platform Protection Profile

All the platform security objectives given in [PP-JCS] are included into this Protection Profile. The security objectives given hereafter are these specifically relevant for the card management and for the SCWS.

5.1.1.1 Basic TOE

This section describes the security objectives for the Basic TOE, applicable to [PPUSIM].

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.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.

Before application can be loaded, the signature provided by the VA is supposed to be verified by the environment.

Communication

O.COMM_AUTH

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

O.COMM_INTEGRITY

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

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, 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).

5.1.2 Java Card System Protection Profile - Open Configuration

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

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. See #.FIREWALL for details.

O.GLOBAL_ARRAYS_CONFID

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

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

O.GLOBAL_ARRAYS_INTEG

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

O.NATIVE

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

O.OPERATE

The TOE must ensure continued correct operation of its security functions. See #.OPERATE for details.

O.REALLOCATION

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

O.RESOURCES

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

5.1.2.3 SERVICES

O.ALARM

The TOE shall provide appropriate feedback information upon detection of a potential security violation. See #.ALARM for details.

O.CIPHER

The TOE shall provide a means to cipher sensitive data for applications in a secure way. In particular, the TOE must support cryptographic algorithms consistent with cryptographic usage policies and standards. See #.CIPHER for details.

O.KEY-MNGT

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

O.PIN-MNGT

The TOE shall provide a means to securely manage PIN objects. See #.PIN-MNGT for details.

Application note:

PIN objects may play key roles in the security architecture of client applications. The way they are stored and managed in the memory of the smart card must be carefully considered, and this applies to the whole object rather than the sole value of the PIN. For instance, the try counter's value is as sensitive as that of the PIN.

O.TRANSACTION

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

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

5.1.2.4 OBJECT DELETION

O.OBJ-DELETION

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

5.1.2.5 APPLET MANAGEMENT

O.DELETION

The TOE shall ensure that both applet and package deletion perform as expected. See #.DELETION for details.

O.LOAD

The TOE shall ensure that the loading of a package into the card is safe. Besides, for code loaded post-issuance, the TOE shall verify the integrity and authenticity evidences generated during the verification of the application package by the verification authority. This verification by the TOE shall occur during the loading or later during the install process.

Application note:

Usurpation of identity resulting from a malicious installation of an applet on the card may also be the result of perturbing the communication channel linking the CAD and the card. Even if the CAD is placed in a secure environment, the attacker may try to capture, duplicate, permute or modify the packages sent to the card. He may also try to send one of its own applications as if it came from the card issuer. Thus, this objective is intended to ensure the integrity and authenticity of loaded CAP files.

O.INSTALL

The TOE shall ensure that the installation of an applet performs as expected (See #.INSTALL for details). Besides, for code loaded post-issuance, the TOE shall verify the integrity and authenticity evidences generated during the verification of the application package by the verification authority. If not performed during the loading process, this verification by the TOE shall occur during the install process.

5.1.2.6 SCP

O.SCP.RECOVERY

If there is a loss of power, or if the smart card is withdrawn from the CAD while an operation is in progress, the SCP must allow the TOE to eventually complete the interrupted operation successfully, or recover to a consistent and secure state.

This security objective refers to the security aspect #.SCP(1): The smart card platform must be secure with respect to the SFRs. Then after a power loss or sudden card removal prior to completion of some communication protocol, the SCP will allow the TOE on the next power up to either complete the interrupted operation or revert to a secure state.

O.SCP.IC

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

This security objective refers to the point (7) of the security aspect #.SCP:

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

5.1.3 (U)SIM

This section defines the security objectives to be achieved by the TOE(U)SIM.

O.Secure_API

The TOE shall provide to application a secure_API means to optimize control on sensitive operations performed by application.

TOE shall provide services for secure array management and to detect loss of data integrity and inconsistent execution flow and react against tearing or fault induction.

O.RND

The TOE must contribute to ensure that random numbers shall not be predictable and shall have sufficient entropy.

O.JCAPI-Services

The TOE must contribute to ensure that data manipulated during SHA and CRC services as defined in [JCAPI301] shall not be observed.

O.REMOTE_SERVICE_AUDIT

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

O.REMOTE_SERVICE_ACTIVATION

The TOE shall perform remote optional platform service activation only when service activation is authorized and only by an authorized actor. Limited to [Gemactivate Administrator (usually Gemalto)] under control of [MNO].

5.1.4 DESFire EV1 Software

O.DESFire_MF-FW DESFIRE Firewall

The TOE shall provide separation between the DESFire EV1 Software and the Security IC Embedded Software. The separation shall comprise software execution and data access.

It is based on Javacard System Firewall already defined in O.FIREWALL, O.GLOBAL_ARRAYS_CONFID, O.GLOBAL_ARRAYS_INTEG, O.NATIVE, O.REALLOCATION, O.OBJ-DELETION.

O.DESFire_MEM-ACCESS Area based Memory Access Control

Access by processor instructions to memory areas is controlled by the TOE. The TOE decides based on the CPU mode (Boot Mode, Test Mode, User Mode) if the requested type of access to the memory area addressed by the operands in the instruction is allowed. It is based on Javacard System Firewall already defined in O.DELETION, O.LOAD,O.INSTALL, the objective O.CARD-MANAGEMENT from GP feature and the O.CONTROLED-ES_LOADING from IC loader feature.

O.DESFire_DATA-ACCESS Access Control to DESFire Data

The TOE must provide an access control mechanism for data stored by the DESFire EV1 Software. The access control mechanism shall apply to read, modify, create and delete operations for data elements and to reading and modifying security attributes as well 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.DESFire_AUTHENTICATION Authentication

The DESFire EV1 Software as part of the TOE must provide an authentication mechanism in order to be able to authenticate authorized users. The authentication mechanism shall be limited to the DESFire EV1 Software and shall be resistant against replay and man-in-the-middle attacks.

O.DESFire_CONFIDENTIALITY Confidential Communication

The TOE must be able to protect the communication of the DESFire EV1 Software by encryption. This shall be implemented by security attributes of the DESFire data element that enforce encrypted communication of the DESFire EV1 Software for the respective data element. During DESFire operation the TOE shall also provide the possibility to detect replay or man-in-the-middle attacks within a session. This shall be implemented by checking verification data sent by the terminal and providing verification data to the terminal.

O.DESFire_TYPE-CONSISTENCY Data type consistency

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

O.DESFire_TRANSACTION Transaction mechanism

The TOE must be able to provide a transaction mechanism that allows updating multiple data elements of the DESFire EV1 Software either all in common or none of them.

O.DESFire_TDES Triple DES Functionality

The TOE shall provide the cryptographic functionality to calculate a Triple DES encryption and decryption to the Security IC Embedded Software and the IC Dedicated Software. The TOE supports directly the calculation of Triple DES with up to three keys. Note: The TOE will ensure the confidentiality of the User Data (and especially cryptographic keys) during Triple DES operation. This is supported by O.Leak-Inherent.

O.DESFire_AES AES Functionality

The TOE shall provide the cryptographic functionality to calculate an AES encryption and decryption to the Security IC Embedded Software and the IC Dedicated Software. The TOE supports directly the calculation of AES with three different key lengths. Note: The TOE will ensure the confidentiality of the User Data (and especially cryptographic keys) during AES operation. This is supported by O.Leak-Inherent.

O.DESFire_SCP_Support

The TOE shall provide access control to objects stored in non volatile memory.

It is based on O.SCP-SUPPORT provided by Javacard system and OS.

O.DESFire_API_GEMACTIVATE_DAP_VERIFICATION

The TOE shall provide access control to load package referencing restricted packages and in particular the DESFIRE API restricting ability to load only if the DAP verification using GemActivate key is successful.

O.DESFire_API_RESTRICTED-ACCESS

The TOE shall provide access control to method of DESFIRE API and DESFIRE shareable interface only if package has been identified using its CODEC AID already registered to DESFIRE EV1.

5.2 Security objectives for the Operational Environment

5.2.1 (U)SIM Java card TM Platform Protection Profile

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.

All the security objectives for the environment of the Java Card System Protection Profile [PP-JCS] are relevant to this Protection Profile except Card Management security objectives which are now part of the TOE.

The specific environment security objectives concerning the (U)SIM Platform are listed below:

5.2.1.1 Basic TOE

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 realisation 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.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.PERSONALIZER

The personalizer shall be a trusted actor in charge of the personalization process. He must ensure the security of the keys it manages and loads into the card:

- Mobile operator keys including OTA keys (telecom keys either generated by the personalizer or by the mobile operator),
- Issuer Security Domain keys (ISD keys),

OE.GEMACTIVATE-ADMIN

The GemActivate administrator shall be a trusted actor responsible for the optional platform service activation in post issuance. The service activation is under the control of the mobile operator as activation is done using OTA communication with MNO OTA servers and associated keys stored in the TOE.

Secure places

OE.PRODUCTION

Production and personalization environment if the TOE delivery occurs before Phase 6 of the TOE life cycle must be trusted and secure.

Policies

Validation and certification

OE.BASIC-APPS-VALIDATION

Standard applications must be analysed 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.

OE.OTA-SERVERS

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

Keys

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.

5.2.2 Java Card System Protection Profile - Open Configuration

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

OE.APPLET

No applet loaded post-issuance shall contain native methods.

OE.VERIFICATION

All the bytecodes shall be verified at least once, before the loading, before the installation or before the execution, depending on the card capabilities, in order to ensure that each bytecode is valid at execution time. See #.VERIFICATION for details.

Additionally, the applet shall follow all the recommendations, if any, mandated in the platform guidance for maintaining the isolation property of the platform.

Application note:

Constraints to maintain the isolation property of the platform are provided by the platform developer in application development guidance. The constraints apply to all application code loaded in the platform.

OE.CODE-EVIDENCE

For application code loaded pre-issuance, evaluated technical measures implemented by the TOE or audited organizational measures must ensure that loaded application has not been changed since the code verifications required in OE.VERIFICATION. For application code loaded post-issuance and verified off-card according to the requirements of OE.VERIFICATION, the verification authority shall provide digital evidence to the TOE that the application code has not been modified after the code verification and that he is the actor who performed code verification. For application code loaded post-issuance and

partially or entirely verified on-card, technical measures must ensure that the verification required in OE.VERIFICATION is performed. On-card bytecode verifier is out of the scope of this Protection Profile.

Application note:

For application code loaded post-issuance and verified off-card, the integrity and authenticity evidence can be achieved by electronic signature of the application code, after code verification, by the actor who performed verification.

5.2.3 (U)SIM

OE.TRUSTED-APPS-DEVELOPER

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

OE.TRUSTED-APPS-PRE-ISSUANCE LOADING

The trusted pre-issuance loading on the platform must be done only using verified applet applying an audited process in a secure environment.

OE.ACTIVATION-KEY-ESCROW

The key escrow is a trusted actor must ensure the security of the keys used for remote service activation during generation, storage, importation in TOE and usage.

5.2.4 DESFire EV1 Software

OE.DESFire_RESP-APPL Treatment of User Data

This means that keys are treated as confidential as soon as they are generated. The keys must be unique with a very high probability, as well as cryptographically strong. For example, if asymmetric algorithms are used, it must be ensured that it is not possible to derive the private key from a related public key using the attacks defined in this Security Target. If keys are imported into the TOE and/or derived from other keys, quality and confidentiality must be maintained. This implies that appropriate key management has to be realized in the environment.

The treatment of User Data is also required when a multi-application operating system is implemented as part of the Security IC Embedded Software on the TOE. In this case the multi-application operating system will not disclose security relevant user data of one application to another application when it is processed or stored on the TOE.

OE.DESFire_Plat-APPL Usage of Hardware Platform

The TOE supports cipher schemes as additional specific security functionality. If required the Security IC Embedded Software shall use these cryptographic services of the TOE and their interface as specified. When key-dependent functions implemented in the Security IC Embedded Software are just being executed, the Security IC Embedded Software must provide protection against disclosure of confidential data (User Data) stored and/or processed in the TOE by using the methods described under Inherent Information Leakage (T.Leak-Inherent) and Forced Information Leakage (T.Leak-Forced). If the Random Number Generator is used for leakage counter-measures, cryptographic operations (e.g. key generation) or cryptographic protocols (e.g. challenge response) these random numbers must be tested appropriately.

OE.DESFire_SECURE-VALUES Generation of secure values

The environment shall generate confidential and secure keys for authentication purpose of the DESFire EV1 Software. 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.DESFire_TERMINAL-SUPPORT Terminal support to ensure integrity and confidentiality

The terminal shall verify information sent by the DESFire EV1 Software 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.

OE.DESFire_API_CODEC-AID-INIT Initialization of AID to access to DESFIRE API

The environment shall manage initialization of the list of authorized AID accessing to DESFIRE API. Such operation is only possible during the personalization in phase 6.

5.3 Security Objectives Rationale

Chapter content has been removed in Public version.

6 Extended requirements

6.1 Extended families

6.1.1 Extended family FCS_RND - Random Number Generation

6.1.1.1 Description

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

6.1.1.2 Extended components

Extended component FCS_RND.1

Description

The generation of random numbers requires that random numbers meet a defined quality metric.

Definition

FCS_RND.1 Random Number Generation

FCS_RND.1.1 The TSF shall provide a mechanism to generate random numbers that meet [assignment: a defined quality metric].

Dependencies: No dependencies.

Rationale

It was chosen to define FCS_RNG.1 explicitly, because Part 2 of the Common Criteria do not contain generic security functional requirements for Random Number generation.

6.1.1.3 Rationale

This family has been introduced initially by IC manufacturer to offer unpredictable random number generation. It is extended here to software platform.

7 Security Functional Requirements

7.1 Security Functional Requirements

7.1.1 (U)SIM Java card TM Platform Protection Profile

This section describes the requirements imposed on the TOE in order to achieve the security objectives laid down in the previous chapter. All the requirements identified in this section are instances of those stated in [CC-2].

The Java Card System Platform security functional requirements are included into this Security Target.

The SFRs listed below state requirements specific to the (U)SIM Platform.

7.1.1.1 Basic TOE

This section describes the SFR for the Basic TOE, applicable to the Security Target.

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

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

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

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

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

FDP_ITC.2.5/CCM The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: **The loading of a new Executable Load File is allowed only if, AID attribute of each dependent Executable File is equal to the identified AID in the CAP File, such AID is unique, SD is personalized and authorized to load. Otherwise, the load of ELF is rejected**.

Application 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.

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 **size of the available memory when the card content management operation starts**.

FDP_UIT.1/CCM Data exchange integrity

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

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

FPT_FLS.1/CCM Failure with preservation of secure state

FPT_FLS.1.1/CCM The TSF shall preserve a secure state when the following types of failures occur: **the Security Domain fails to load/install an Executable File / application instance as described in GP22 §9.3.5**.

Security Domain

FDP_ACC.1/SD Subset access control

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

Subjects: S.INSTALLER, S.ADEL, S.CAD (from [PP-JCS]) and S.SD

Objects: Delegation Token, DAP Block and Load File

Operations: GlobalPlatform's card content management APDU commands and API methods.

FDP_ACF.1/SD Security attribute based access control

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

Subjects:

S.INSTALLER, defined in [PP-JCS] and represented by the GlobalPlatform Environment (OPEN) on the card, the Card Life Cycle attributes (defined in Section 5.1.1 of [GP]);

S.ADEL, also defined in [PP-JCS] and represented by the GlobalPlatform Environment (OPEN) on the card;

S.SD receiving the Card Content Management commands (through APDUs or APIs) with a set of privileges (defined in Section 6.6.1 of [GP]), a life-cycle status (defined in Section 5.3.2 of [GP]) and a Secure Communication Security level (defined in Section 10.6 of [GP]);

S.CAD, defined in [PP-JCS], the off-card entity that communicates with the S.INSTALLER through S.SD;

Objects:

The Delegation Token, in case of Delegated Management operations, with the attributes Present or Not Present;

The DAP Block, in case of application loading, with the attributes Present or Not Present;

The Load File or Executable File, in case of application loading, installation, extradition or registry update, with a set of intended privileges and its targeted associated SD AID.

the following security attributes:

The Default Selected attribute specifies whether the applet instance is the one that should be executed when no application has been explicitly selected.

The Application State attribute specifies the current life cycle state of the application instance, which may be either SELECTABLE, APPLICATION_SPECIFIC,LOCKED.

The Card State attribute, is the current state in the life cycle of the card, which may be either OP_READY, INITIALIZED, SECURED, CARD_LOCKED, or TERMINATED.

The Card Lock attribute specifies whether the applet is allowed to temporary lock the services of the smart card.

The Card Termination attribute specifies whether the applet is allowed to definitely disable the services of the smart card.

The Registered Applications attribute specifies the Executable Files and application instances that have been installed on the card so far and their dependencies.

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

Runtime behavior rules defined by GlobalPlatform for:

loading (Section 9.3.5 of [GP]);

installation (Section 9.3.6 of [GP]);

extradition (Section 9.4.1 of [GP]);

registry update (Section 9.4.2 of [GP]);

content removal (Section 9.5 of [GP]).

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

Rule SD-1: A card administration request may be accepted only if the APDU command specifying the request is well-formed according to [GP22].

Rule SD-2: A card administration request other than requesting card management data may be accepted only if the Card State is not TERMINATED.

Rule SD-3: The selection of an applet instance may be accepted only if the Applet State is not LOCKED.

Rule SD-4: The update of the life cycle state of an application instance is accepted only if the new state is consistent with its current life cycle state according to GlobalPlatform's life cycle rules (either coming from an APDU command or from an application instance through the GP API).

Rule SD-5: A request for installing an Executable Load File may be accepted only if there is enough resources for loading the Executable File, and no Executable File on the card has been already registered with the specified AID.

Rule SD-6: A Executable Load File block may be loaded only if all its previous blocks have been received in order, and there are sufficient resources for storing the new one.

Rule SD-7: A new applet instance may be created only if the Package Properties enables applet instantiation or multiple applet instances (if there is already an instance for that applet) but also if the AID specified for the applet instance is not already used for another applet or Executable File installed on the card, and the privileges specified for it are consistent with the GlobalPlatform rules specified in [VGP].

Rule SD-8: An Executable File may be deleted from the smart card only if it is not reachable from other Executable Files or application instances on the card.

Rule SD-9: An applet instance may be deleted from the card only it is not currently active on a logical channel, and none of the resources it has allocated is reachable from other Executable Files or Application instances installed on the card.

Rule SD-10: An applet instance may lock the card only if it has the Card Lock privilege.

Rule SD-11: An applet instance may terminate the card only if it has the Card Termination privilege.

Rule SD-12: An applet instance may unlock the CVM service or modify the CVM try limit or PIN code only if it has the CVM privilege.

Rule SD-13: A request involving the use of any of the Security domain keys is accepted only if the concerned keys are integer.

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

FMT_MSA.1/SD Management of security attributes

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

Application state of an application instance (1)

Default selected application (2)

Card Life cycle state (3)

Package properties (4)

Application association (5)

to

the Security Domain and the application instance itself (1)

the Security Domain (2&4)

the Security Domain and application with privilege (Card Lock or Terminated)(3).

FMT_MSA.3/SD Static attribute initialisation

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

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

Refinement:

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

The Default Selected application shall be the ISD.

The initial value of the Application State of an applet instance shall be SELECTABLE.

Application note:

When the TOE enters the life cycle phases under the scope of this Security Target, the Card State shall be at least SECURED.

The initial value of the Application State of an applet instance shall be SELECTABLE.

The initial Package Properties shall enable all card content management operations on the package.

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

FMT_SMF.1/SD Specification of Management Functions

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

Restricting the properties associated to a given package

Registering a new Executable File or application instance in the GP registry.

Removing the specified entries from the GP registry when a DELETE command is received.

Unsetting it as the Default Select application and set this privilege to a new application instance.

Granting the privileges that the authorized entities (MNO, or Application Provider) specifies when a new application instance is installed.

FMT_SMR.1/SD Security roles

FMT_SMR.1.1/SD The TSF shall maintain the roles

Issuer Security Domain

Supplementary Security Domain

Certification Authority Security Domain.

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

Secure Channel

FCO_NRO.2/SC Enforced proof of origin

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

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

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

FDP_IFC.2/SC Complete information flow control

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

the subjects S.CAD and S.SD, involved in the exchange of messages between the (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.

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

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

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

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

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

FDP_IFF.1/SC Simple security attributes

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

Subjects:

S.SD receiving the Card Content Management commands (through APDUs or APIs). This subject can be the ISD, an APSD.

S.CAD the off-card entity that communicates with the S.SD.

Information:

load file, in case of application loading;

applications or SD privileges, in case of application installation or registry update;

personalization keys and/or certificates, in case of application or SD personnalization.

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

The Challenge is a random number generated by the subject in order to identify the current session.

The Cryptogram is a secret relative to the current smart card session that serves to authenticate the on- and off-card subjects. The cryptogram is derived from the challenges of both the card and the terminal.

The Key Set is a collection of three keys (Secure Channel Encryption Key (SENC), a Command Message Authentication Code Key (C-MAC) and a Data Encryption Key (DEK)) used to encrypt the Derivation Data in order to generate the session keys. It is identified by a key version number.

-- The Session Keys is a set of keys derived from KeySet and sequence counter to be used to verify the origin and integrity of the received message, and to decrypt their contents. This set is made of the following keys: * Command Message Authentication Code Key (C-MAC session key); * Encryption Key (S-ENC session key); * Data Encryption Key (DEK session key).

-- The Command Security Level defined for the messages that the card receives through the secure channel. The possible security levels are: NO-SEC (clear text), C-AUTHENTICATED (authentication of the command's issuer), C-MAC (authentication of the issuer and integrity of the command), C-DEC (authentication of the issuer, integrity and confidentiality of the command).

-- The Initial Chaining Vector (ICV) is a value used to compute the MAC value of a message, which relates it to the previous messages of the current session.

The security attributes for SCP80 are:

CPL, CHL giving Information about the length of the received message and the length of the security header in that message;

SPI containing the security level applied to the incoming message, and response (if any), defining properties for message integrity and authentication, replay detection and sequence integrity and confidentiality;

TAR (Target Application Reference), indicating the application which the message is addressed to;

KIC and KID both contain information on the keys (key set number and the key algorithm) to be used when checking the security;

CNTR, a synchronization counter to avoid playing the same message several times;

PCNTR, padding information used only when the message is encrypted;

and finally a signature, that can be either a basic CRC of the message or a signature involving keys.

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

Runtime behavior rules defined by GlobalPlatform for:

loading (Section 9.3.5 of [GP]);

installation (Section 9.3.6 of [GP]);

extradition (Section 9.4.1 of [GP]);

registry update (Section 9.4.2 of [GP]);

SD personalization rules, pull and push models (Section 11 of [GP-UICC]).

Rule IFF-1: The SD may process a RECEIVE (INITIALIZE-UPDATE) operation only if the key set specified in the command exist in the SD and is integer.

Rule IFF-2: The ISD may process a RECEIVE (EXTERNAL-AUTHENTICATE) operation if the following conditions hold:

The cryptogram received from the off-card subject is equal to the cryptogram computed by the Security Domain.

The MAC attached to the message has been generated from the CMAC session key and the current value of the ICV.

Rules IFF-3: The ISD may process a RECEIVE (GET-DATA) operation if the following condition holds: If the command security level is at least C-

MAC, the MAC attached to the message has been generated from the command using the C-MAC session key and the current value of the ICV.

Rules IFF-4:The ISD may process a RECEIVE (M) operation for any other command M different from the ones cited in the rules above if the following conditions hold:

The current security level is at least AUTHENTICATED.

If the command security level is at least C-MAC, the MAC attached to the message has been generated from the clear-text command using the C-MAC session key and the current value of the ICV.

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

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

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

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

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 [GP] 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**.

FMT_MSA.1/SC Management of security attributes

FMT_MSA.1.1/SC The TSF shall enforce the **Secure Channel Protocol (SCP) information flow control policy** to restrict the ability to **modify** the security attributes **(1) key set, Static keys, Command security Level, Secure channel protocol of a security domain (2) Session Keys, Sequence Counter and ICV of a session (for SCP02) (3) SPI, TAR, CNTR, PCNTR, signature (for SCP80)** to **(1 & 2 & 3) the actor associated with the security domain:**

The Mobile Network Operator for ISD,

The application Provider for SSD.

Application note:

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

FMT_MSA.3/SC Static attribute initialisation

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

FMT_MSA.3.2/SC The TSF shall allow the **authorized entities (MNO, or Application Provider)** to specify alternative initial values to override the default values when an object or information is created.

Refinement:

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

FMT_SMF.1/SC Specification of Management Functions

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

Management functions specified in GlobalPlatform specifications [GP]:

loading (Section 9.3.5 of [GP]);

installation (Section 9.3.6 of [GP]);

extradition (Section 9.4.1 of [GP]);

registry update (Section 9.4.2 of [GP]);

SD personalization rules, pull and push models (Section 11 of [GP-UICC]).

The management functions are:

(for SCP02)

Generating a new card challenge during the set up of a Secure Channel.

Generating the session keys for the Secure Channel from the specified static key set and its associated Sequence Counter.

Generating the card cryptogram from the host and card challenges and the session keys.

Increasing by one the Sequence Counter associated to the specified Key Set upon successful opening a Secure Channel.

Setting the security level of the Secure Channel as the authenticated authorized entities (MNO, or Application Provider) had specified during its set up.

Updating the current value of the ICV upon reception of a new message through the Secure Channel.

On request of the Issuer or authorized application provider, loading or replacing the static keys that the associated Security Domain uses to open a Secure Channel.

(For SCP80):

modifying parameter values (CPL, CHL, SPI, Kic, Kid, TAR, CNTR, PCNTR, signature),

Setting the security level of the Secure Channel as the authenticated authorized entities (MNO, or Application Provider) had specified during its set up,

On request of the Issuer or authorized application provider, loading or replacing the static keys that the associated Security Domain uses to open a Secure Channel.

Application note:

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

FTP_ITC.1/SC Inter-TSF trusted channel

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

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

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

loading or deleting an Executable Load file;

installing or removing an application instance;

extrading an Executable Load file or an application instance;

registry update;

Loading or removing a KeySet;

SD personalization;

changing the Application Life Cycle or card Life Cycle;.

7.1.2 Java Card System Protection Profile - Open Configuration

This section states the security functional requirements for the Java Card System - Open configuration. For readability and for compatibility with the original Java Card System Protection Profile Collection - Standard 2.2 Configuration [PP/0305], requirements are arranged into groups. All the groups defined in the table below apply to this Protection Profile.

Group	Description
Core with Logical Channels (*CoreG_LC*)	**The CoreG_LC contains the requirements concerning the runtime environment of the Java Card System implementing logical channels. This includes the firewall policy and the requirements related to the Java Card API. Logical channels are a Java Card specification version 2.2 feature. This group is the union of requirements from the Core (*CoreG*) and the Logical channels (*LCG*) groups defined in [PP/0305] (cf. Java Card System Protection Profile Collection [PP-JCS]).**
Installation (*InstG*)	**The InstG contains the security requirements concerning the installation of post-issuance applications. It does not address card management issues in the broad sense, but only those security aspects of the installation procedure that are related to applet execution.**
Applet deletion (*ADELG*)	**The ADELG contains the security requirements for erasing installed applets from the card, a feature introduced in Java Card specification version 2.2.**
Object deletion (*ODELG*)	**The ODELG contains the security requirements for the object deletion capability. This provides a safe memory recovering mechanism. This is a Java Card specification version 2.2 feature.**
Secure carrier (*CarG*)	**The CarG group contains minimal requirements for secure downloading of applications on the card. This group contains the security requirements for preventing, 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), 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 ([JCRE301], §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.
S.CAD	The CAD represents the actor that requests, by issuing commands to the card. It also plays the role of the off-card entity that communicates with the S.INSTALLER.
S.INSTALLER	The installer is the on-card entity which acts on behalf of the card issuer. This subject is involved in the loading of packages and installation of applets.
S.JCRE	The runtime environment under which Java programs in a smart card are executed.
S.JCVM	The bytecode interpreter that enforces the firewall at runtime.
S.LOCAL	Operand stack of a JCVM frame, or local variable of a JCVM frame containing an object or an array of references.
S.MEMBER	Any object's field, static field or array position.
S.PACKAGE	A package is a namespace within the Java programming language that may contain classes and interfaces, and in the context of Java Card technology, it defines either a user library, or one or several applets.

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

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

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

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

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

Security attribute	Description/Value
Active Applets	The set of the active applets' AIDs. An active applet is an applet that is selected on at least one of the logical channels.
Applet Selection Status	"Selected" or "Deselected".
Applet's version number	The version number of an applet (package) indicated in the export file.
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 ([JCVM301], §4.5.2).
LC Selection Status	Multiselectable, Non-multiselectable or "None".
LifeTime	CLEAR_ON_DESELECT or PERSISTENT (*).
Owner	The Owner of an object is either the applet instance that created the object or the package (library) where it has been defined (these latter objects can only be arrays that initialize static fields of the package).
Package AID	The AID of each package indicated in the export file.
Registered Applets	The set of AID of the applet instances registered on the card.
Resident Packages	The set of AIDs of the packages already loaded on the card.
Selected Applet Context	Package AID or "None".
Sharing	Standards, SIO, Java Card RE entry point or global array.
Static References	Static fields of a package may contain references to objects. The Static References attribute records those references.

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

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.INSTANCE_FIELD(O.JAVAOBJECT, field)	**Read/Write a field of an instance of a class in the Java programming language.**
OP.INVK_VIRTUAL(O.JAVAOBJECT, method, arg1,...)	**Invoke a virtual method (either on a class instance or an array object).**
OP.INVK_INTERFACE(O.JAVAOBJECT, method, arg1,...)	**Invoke an interface method.**
OP.JAVA(...)	**Any access in the sense of [JCRE301], §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.THROW(O.JAVAOBJECT)	**Throwing of an object (athrow, see [JCRE301], §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 ([JCRE301], §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 ([JCRE301], §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 ([JCRE301], §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 ([JCRE301], §6.2.8.6): S.PACKAGE may perform OP.INVK_INTERFACE upon an O.JAVAOBJECT whose Sharing attribute has the value "SIO", and whose Context attribute has the value "Package AID", only if the invoked interface method extends the Shareable interface and one of the following conditions applies:

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

b) The value of the attribute Selection Status of the package whose AID is "Package AID" is "Non-multiselectable", and either "Package AID" is the value of the currently selected applet or otherwise "Package AID" does not occur in the attribute Active Applets.

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

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

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

The Sharing attribute defines four categories of objects:

Standard ones, whose both fields and methods are under the firewall policy,

Shareable interface Objects (SIO), which provide a secure mechanism for inter-applet communication,

JCRE entry points (Temporary or Permanent), who have freely accessible methods but protected fields,

Global arrays, having both unprotected fields (including components; refer to JavaCardClass discussion above) and methods.

When a new object is created, it is associated with the Currently Active Context. But the object is owned by the applet instance within the Currently Active Context when the object is instantiated ([JCRE301], §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).

([JCRE301], 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.

([JCRE301], §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 ([JCVM301], §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. ([JCRE301], §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 **[No additional rules]**.

FDP_IFF.1.4/JCVM The TSF shall explicitly authorise an information flow based on the following rules: **[No additional rules]**.

FDP_IFF.1.5/JCVM The TSF shall explicitly deny an information flow based on the following rules: **[No additional rules]**.

Application note:

The storage of temporary Java Card RE-owned objects references is runtime-enforced ([JCRE301], §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 [JVM], §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 [JCRE301], §4 and [JCVM301], §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 [JCRE301], §4 and [JCVM301], §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 initialisation

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

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

Application note:

FMT_MSA.3.1/FIREWALL

Objects' security attributes of the access control policy are created and initialized at the creation of the object or the subject. Afterwards, these attributes are no longer mutable (FMT_MSA.1/JCRE). At the creation of an object (OP.CREATE), the newly created object, assuming that the FIREWALL access control SFP permits the operation, gets its Lifetime and Sharing attributes from the parameters of the operation; on the contrary, its Context attribute has a default value, which is its creator's Context attribute and AID respectively ([JCRE301], §6.1.3). There is one default value for the Selected Applet Context that is the default applet identifier's Context, and one default value for the Currently Active Context that is "Java Card RE".

The knowledge of which reference corresponds to a temporary entry point object or a global array and which does not is solely available to the Java Card RE (and the Java Card virtual machine).

FMT_MSA.3.2/FIREWALL

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

FMT_MSA.3/JCVM Static attribute initialisation

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

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

FMT_SMF.1 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/DES Cryptographic key generation

FCS_CKM.1.1/DES The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **TDES Key generation** and specified cryptographic key sizes **112 bits for TDES 2 keys, 168 bits for TDES 3 keys** that meet the following: **none (random numbers generation)**.

Application note:

The keys can be generated and diversified in accordance with [JCAPI222] standard in classes KeyBuilder.

FCS_CKM.1/AES Cryptographic key generation

FCS_CKM.1.1/AES The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **AES Key generation** and specified cryptographic key sizes **128 bits for AES key** that meet the following: **none (random numbers generation)**.

Application note:

The keys can be generated and diversified in accordance with [JCAPI222] standard in classes KeyBuilder.

FCS_CKM.1/RSA Cryptographic key generation

FCS_CKM.1.1/RSA The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **[see application note]** and specified cryptographic key sizes **[1536 to 2048 bits]with CRT** that meet the following: **[see application note]**.

Application note:

The keys can be generated and diversified in accordance with [JCAPI222] standard in classes KeyBuilder and KeyPair (at least Session key generation.

FCS_CKM.2/DES Cryptographic key distribution

FCS_CKM.2.1/DES The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method **(see application note)** that meets the following: **(see application note)**.

Application note:

Command SetKEY that meets [JCAPI222] standard.

FCS_CKM.2/AES Cryptographic key distribution

FCS_CKM.2.1/AES The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method **(see application note)** that meets the following: **(see application note)**.

Application note:

Command SetKEY that meets [JCAPI222] standard.

FCS_CKM.2/RSA Cryptographic key distribution

FCS_CKM.2.1/RSA The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method **(see application note)** that meets the following: **(see application note)**.

Application note:

Command SetKEY that meets [JCAPI222] standard.

FCS_CKM.3/DES Cryptographic key access

FCS_CKM.3.1/DES The TSF shall perform **(see application note)** in accordance with a specified cryptographic key access method **(see application note)** that meets the following: **(see application note)**.

Application note:

The keys can be accessed in accordance with [JCAPI222] standard in class Key.

FCS_CKM.3/AES Cryptographic key access

FCS_CKM.3.1/AES The TSF shall perform **(see application note)** in accordance with a specified cryptographic key access method **(see application note)** that meets the following: **(see application note)**.

Application note:

The keys can be accessed in accordance with [JCAPI222] standard in class Key.

FCS_CKM.3/RSA Cryptographic key access

FCS_CKM.3.1/RSA The TSF shall perform **(see application note)** in accordance with a specified cryptographic key access method **(see application note)** that meets the following: **(see application note)**.

Application note:

The keys can be accessed in accordance with [JCAPI222] standard in class Key.

FCS_CKM.4 Cryptographic key destruction

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method **(see application note)** that meets the following: **(see application note)**.

Application note:

The keys are reset as specified in [JCAPI222] Key class, with the method clearKey(). Any access to a cleared key for ciphering or signing shall throw an exception.

FCS_COP.1/DES_CIPHER Cryptographic operation

FCS_COP.1.1/DES_CIPHER The TSF shall perform **[encryption and decryption of applet instance's data]** in accordance with a specified cryptographic algorithm **[Triple DES either in CBC or ECB mode and with padding schme (NOPAD,ISO9797 or PKCS#5)]** and cryptographic key sizes **[112 or 168 bits]** that meet the following: **[FIPS PUB 46-3, FIPS PUB 81, ISO 9797, according to JCAPI222]**.

Application note:

The TOE shall provide a subset of cryptographic operations defined in [JCAPI222] (see javacardx.crypto.Cipher and javacardx.security packages).

FCS_COP.1/DES_MAC_COMP Cryptographic operation

FCS_COP.1.1/DES_MAC_COMP The TSF shall perform **[MAC generation or verification of applet instance's data]** in accordance with a specified cryptographic algorithm **[Triple DES in CBC mode and with or without padding generating MAC on 4-bytes or 8-bytes]** and cryptographic key sizes **[112 or 168 bits]** that meet the following: **[FIPS PUB 46-3, FIPS PUB 81, ISO 9797, according to JCAPI222]**.

Application note:

The TOE shall provide a subset of cryptographic operations defined in [JCAPI222] (see javacardx.crypto.Cipher and javacardx.security packages).

FCS_COP.1/AES_CIPHER Cryptographic operation

FCS_COP.1.1/AES_CIPHER The TSF shall perform **[encryption and decryption of applet instance's data]** in accordance with a specified cryptographic algorithm **[AES (128 bits) either in CBC or ECB mode without padding]** and cryptographic key sizes **[128 bits]** that meet the following: **[FIPS PUB 197, FIPS PUB 81, ISO 9797, according to JCAPI222]**.

Application note:

The TOE shall provide a subset of cryptographic operations defined in [JCAPI222] (see javacardx.crypto.Cipher and javacardx.security packages).

FCS_COP.1/AES_MAC_COMP Cryptographic operation

FCS_COP.1.1/AES_MAC_COMP The TSF shall perform **[MAC generation or verification of applet instance's data]** in accordance with a specified cryptographic algorithm **[AES (128bits) in CBC mode and with or without padding generating MAC on 4-bytes or 8-bytes]** and cryptographic key sizes **[112 or 168 bits]** that meet the following: **[FIPS PUB 197, FIPS PUB 81, ISO 9797, according to JCAPI222]**.

Application note:

The TOE shall provide a subset of cryptographic operations defined in [JCAPI222] (see javacardx.crypto.Cipher and javacardx.security packages).

FCS_COP.1/RSA_SIGN Cryptographic operation

FCS_COP.1.1/RSA_SIGN The TSF shall perform **[signature generation or verification of applet instance's data]** in accordance with a specified cryptographic algorithm **[RSA with CRT in mode ISO 148888 with padding scheme (ISO9796 or PKCS #1)]** and cryptographic key sizes **[1536 to 2048 bits]** that meet the following: **[PKCS #1 Version 2.1]**.

Application note:

The TOE shall provide a subset of cryptographic operations defined in [JCAPI222] (see javacardx.crypto.Cipher and javacardx.security packages).

FCS_COP.1/RSA_CIPHER Cryptographic operation

FCS_COP.1.1/RSA_CIPHER The TSF shall perform **[encryption or decryption of applet instance's data]** in accordance with a specified cryptographic algorithm **[RSA with CRT]** and cryptographic key sizes **[1536 to 2048 bits]** that meet the following: **[PKCS #1 Version 2.1]**.

Application note:

The TOE shall provide a subset of cryptographic operations defined in [JCAPI22] (see javacardx.crypto.Cipher and javacardx.security packages).

FCS_COP.1/HMAC Cryptographic operation

FCS_COP.1.1/HMAC The TSF shall perform **[computation of a hash value for applet instance's data]** in accordance with a specified cryptographic algorithm **[HMAC, HMAC MD5, HMAC SHA-384 (48 bytes) or HMAC SHA-256 (32 bytes),HMAC SHA-224 and HMAC SHA-1]** and cryptographic key sizes **[4-64 bytes]** that meet the following: **[rfc2104 & 2085 & 3874]**.

Application note:

The TOE shall provide a subset of cryptographic operations defined in [JCAPI222] (see javacardx.crypto.Cipher and javacardx.security packages).

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 [JCRE301], §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 [JCAPI222].

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 [JCRE301], §5.1 and §3.6.1.

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 ([JCRE301], §4.2.

Moreover in [JCRE301]§3.6.1, Transient data of CLEAR_ON_DESELECT objects associated with each applet instance that was active on a logical channel over the contactless I/O interface and that does not have an applet instance from the same package active on any logical channel over the contacted I/O interface, is reset to the default value.

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 [JCRE301], §7.7, within the bounds of the Commit Capacity ([JCRE301], §7.8), and those described in [JCAPI222]**.

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 [JCAPI222] (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(), [JCAPI222] and ([JCRE301], §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: **integrityCheckData**.

FDP_SDI.2.2 Upon detection of a data integrity error, the TSF shall **increase a counter of integrity error event and mute the card if counter is greater than max value**.

Application note:

The following data persistently stored by TOE have a integrity check data security attribute:

* PIN (objects instance of class OwnerPin), * Key (i.e. objects instance of classes implemented the interface Key), * package.

FPR_UNO.1 Unobservability

FPR_UNO.1.1 The TSF shall ensure that **[any user]** are unable to observe the operation **[read, write, cryptographic operations]** on **[PIN, Key]** by **[any other user or subject]**.

Application note:

Although it is not required in [JCRE301] specifications, the non-observability of operations on sensitive information such as keys appears as impossible to circumvent in the smart card world.

FPT_FLS.1/JCS Failure with preservation of secure state

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

Application note:

The Java Card RE Context is the Current context when the Java Card VM begins running after a card reset ([JCRE301], §6.2.3) or after a proximity card (PICC) activation sequence ([JCRE301]). Behavior of the TOE on power loss and reset is described in [JCRE301], §3.6 and §7.1. Behavior of the TOE on RF signal loss is described in [JCRE301], §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 [JCVM301] specification,

the API tokens defined in the export files of reference implementation,

when interpreting the TSF data from another trusted IT product.

Application note:

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

AID Management

FIA_ATD.1/AID User attribute definition

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

Package AID,

Applet's version number,

Registered applet AID,

Applet Selection Status ([JCVM301], §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, active context**.

FIA_USB.1.2/AID The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users: **Package AID are defined with associated value during loading and with context identifier**.

FIA_USB.1.3/AID The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: **[None]**.

Application note:

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

FMT_MTD.1/JCRE Management of TSF data

FMT_MTD.1.1/JCRE The TSF shall restrict the ability to **modify** the **list of registered applets' AIDs** to **the JCRE**.

Application note:

The installer and the Java Card RE manage other TSF data such as the applet life cycle or CAP files, but this management is implementation specific. Objects in the Java programming language may also try to query AIDs of installed applets through the lookupAID(...) API method.

The installer, applet deletion manager or even the card manager may be granted the right to modify the list of registered applets' AIDs in specific implementations (possibly needed for installation and deletion; see #.DELETION and #.INSTALL).

FMT_MTD.3/JCRE Secure TSF data

FMT_MTD.3.1/JCRE The TSF shall ensure that only secure values are accepted for **the registered applets' AIDs**.

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 [PP-JCS], 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 ([JCVM301], §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 [JCVM301] 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 ([JCVM301], §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 ([JCVM301], §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" ([JCVM301]) 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 ([JCVM301], §4.4).

The installation (the invocation of an applet's install method by the installer) is implementation dependent ([JCRE301], §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, [JCRE301], §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 [JCRE301] §11.1.5**.

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 **a failure or service discontinuity** is not possible, the TSF shall enter a maintenance mode where the ability to return to a secure state is provided.

FPT_RCV.3.2/Installer For **[detection of a potential loss of integrity during the transmission of an Executable Load File to the card, abortion of the installation process of an Executable Load File, or any fatal error occurred during the linking of an Executable Load File to the Executable Files already installed on the card]**, the TSF shall ensure the return of the TOE to a secure state using automated procedures.

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

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

Application note:

FPT_RCV.3.1/Installer:

This element is not within the scope of the Java Card specification, which only mandates the behavior of the Java Card System in good working order. Further details on the "maintenance mode" shall be provided in specific implementations. The following is an excerpt from [CC-2], p298: In this maintenance mode normal operation might be impossible or severely restricted, as otherwise insecure situations might occur. Typically, only authorised users should be allowed access to this mode but the real details of who can access this mode is a function of FMT: Security management. If FMT: Security management does not put any controls on who can access this mode, then it may be acceptable to allow any user to restore the system if the TOE enters such a state. However, in practice, this is probably not desirable

as the user restoring the system has an opportunity to configure the TOE in such a way as to violate the SFRs.

FPT_RCV.3.2/Installer:

Should the installer fail during loading/installation of a package/applet, it has to revert to a "consistent and secure state". The Java Card RE has some clean up duties as well; see [JCRE301], §11.1.5 for possible scenarios. Precise behavior is left to implementers. This component shall include among the listed failures the deletion of a package/applet. See ([JCRE301], 11.3.4) for possible scenarios. Precise behavior is left to implementers.

Other events such as the unexpected tearing of the card, power loss, and so on, are partially handled by the underlying hardware platform (see [PP0035]) and, from the TOE's side, by events "that clear transient objects" and transactional features. See FPT_FLS.1.1, FDP_RIP.1/TRANSIENT, FDP_RIP.1/ABORT and FDP_ROL.1/FIREWALL.

FPT_RCV.3.3/Installer:

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

7.1.2.3 ADELG Security Functional Requirements

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

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

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 an object O' that is reachable according to either (1) or (2).

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

R.JAVA.14 ([JCRE301], §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.

R.JAVA.15 ([JCRE301], §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.

R.JAVA.16 ([JCRE301], §11.3.4.2, Applet/Library Package Deletion): S.ADEL may perform OP.DELETE_PCKG upon an O.CODE_PKG only if,

(1) S.ADEL is currently selected,

(2) no reachable O.JAVAOBJECT, from a package distinct from O.CODE_PKG that is an instance of a class that belongs to O.CODE_PKG, exists on the card and

(3) there is no resident package on the card that depends on O.CODE_PKG.

R.JAVA.17 ([JCRE301], §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.

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

FDP_ACF.1.4/ADEL [Editorially Refined] The TSF shall explicitly deny access of **any subject but S.ADEL to O.CODE_PKG or O.APPLET for the purpose of deleting them from the card**.

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 [JCRE301], §11.3.4.1, §11.3.4.2 and §11.3.4.3.

FMT_MSA.1/ADEL Management of security attributes

FMT_MSA.1.1/ADEL The TSF shall enforce the **ADEL access control SFP** to restrict the ability to **modify** the security attributes **Registered Applets and Resident Packages** to **the Java Card RE**.

FMT_MSA.3/ADEL Static attribute initialisation

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

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

FMT_SMF.1/ADEL Specification of Management Functions

FMT_SMF.1.1/ADEL The TSF shall be capable of performing the following management functions: **modify the list of registered applets' AIDs and the Resident Packages**.

Application note:

The modification of the Active Applets security attribute should be performed in accordance with the rules given in [JCRE301], §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 [JCRE301], §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 ([JCRE301], §11.3.4.)

7.1.2.4 ODELG Security Functional Requirements

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

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 [JCAPI222].

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).

7.1.2.5 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 **as assumption the key used is kept integer and confidential by origin**.

Application note:

FCO_NRO.2.1/CM:

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

FCO_NRO.2.3/CM:

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

FDP_IFC.2/CM Complete information flow control

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

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

Application note:

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

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

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

FDP_IFF.1/CM Simple security attributes

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

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

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

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

an (INITIALIZE-UPDATE) operation only if the key set specified in the command exist,

an (EXTERNAL-AUTHENTICATE) operation if the following conditions are fulfilled: 1) The cryptogram received from the off-card subject is equal to the cryptogram computed by the Security Domain. 2) The MAC attached to the message has been generated using the CMAC session key and the current value of the ICV.

a (GET-DATA) operation if the following condition are fulfilled: 1) If the command security level is at least C-MAC, 2) the MAC attached to the message has been generated from the command using the C-MAC session key and the current value of the ICV.

any received operation for any other command if the following conditions hold: 1) The current security level is at least AUTHENTICATED. 2) If the command security level is at least C-MAC, the MAC attached to the message has been generated from the clear-text command using the C-MAC session key and the current value of the ICV.

FDP_IFF.1.5/CM The TSF shall explicitly deny an information flow based on the following rules: **[A Security Domain may always process a (SELECT) operation or a (Get DATA) operation at the security level NO-SEC]**.

Application note:

FDP_IFF.1.1/CM:

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

FDP_IFF.1.2/CM:

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

FDP_UIT.1/CM Data exchange integrity

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

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

Application note:

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

FIA_UID.1/CM Timing of identification

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

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

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 **[Card Life cycle, Security Level]** to **[Card Manager]**.

FMT_MSA.3/CM Static attribute initialisation

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

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

FMT_SMF.1/CM Specification of Management Functions

FMT_SMF.1.1/CM The TSF shall be capable of performing the following management functions: **[modification of the Card life cycle inducing availability of management functions]**.

FMT_SMR.1/CM Security roles

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

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

FTP_ITC.1/CM Inter-TSF trusted channel

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

FTP_ITC.1.2/CM [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 SCP

This section states the security functional requirements for the Smart Card Platform.

Operating System

This section presents those requirements of the Smart Card Platform group that concern the Operating System. Due to enlargement in the scope of evaluation, the requirements related to OS are now assigned to the TOE and no more to the environment. Other internal security mechanisms are not addressed by SFR but ADV_ARC activities.

FPT_RCV.3/OS Automated recovery without undue loss

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

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

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

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

FPT_RCV.4/OS Function recovery

FPT_RCV.4.1/OS 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.

Integrated Circuit

The section should contain the requirements of the Smart Card Platform group introduced in [JCSPP] concerning the Integrated Circuit. Due to enlargement in the scope of evaluation, the requirements related to IC are now assigned to the TOE and no more to the environment.

Those requirements are fulfilled in the [ICST] and are covered by the IC certificate reused in the composite evaluation process. There are not repeated here.

They mainly concern protecting the smart card's chip against physical tampering, preventing the disclosure of information when it is transferred from different physical parts of the chip, providing the basic DES operation, and keeping a secure state when a malfunction is detected and providing an independent security domain for the hardware.

7.1.3 (U)SIM

7.1.3.1 Crypto JCAPI

FCS_COP.1/SHA2 Cryptographic operation

FCS_COP.1.1/SHA2 The TSF shall perform **[computation of a hash value for applet instance's data]** in accordance with a specified cryptographic algorithm **[SHA-384 (48 bytes) or SHA-256 (32 bytes) or SHA2-224 (32 bytes)]** and cryptographic key sizes **[None]** that meet the following: **[FIPS 180-3]**.

Application note:

The TOE shall provide a subset of cryptographic operations defined in [JCAPI222] (see javacardx.crypto.Cipher and javacardx.security packages).

FCS_COP.1/CRC Cryptographic operation

FCS_COP.1.1/CRC The TSF shall perform **[Computation of checksum CRC16 or CRC32 of applet instance's data]** in accordance with a specified cryptographic algorithm **[CRC16 or CRC32]** and cryptographic key sizes **[none]** that meet the following: **[ISO3309]**.

Application note:

The TOE shall provide a subset of cryptographic operations defined in [JCAPI222] (see javacardx.crypto.Cipher and javacardx.security packages).

FCS_RND.1 Random Number Generation

FCS_RND.1.1 The TSF shall provide a mechanism to generate random numbers that meet **the STANDARD level specified in [DCSSI2741]**.

7.1.3.2 SecureAPI

FPT_FLS.1/SecureAPI Failure with preservation of secure state

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

FPT_ITT.1/SecureAPI Basic internal TSF data transfer protection

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

FPR_UNO.1/SecureAPI Unobservability

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

7.1.3.3 GemActivate

FMT_SMR.1/GemActivate Security roles

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

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

FMT_SMF.1/GemActivate Specification of Management Functions

FMT_SMF.1.1/GemActivate The TSF shall be capable of performing the following management functions: **activation of optional platform service**.

FMT_MOF.1/GemActivate Management of security functions behaviour

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

FMT_MSA.1/GemActivate Management of security attributes

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

FMT_MTD.1/GemActivate Management of TSF data

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

FDP_ACC.1/ GemActivate_DAP Subset access control

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

Subjects: S.INSTALLER, S.GEMACTIVATE and S.SD

Objects: GemActivate DAP Block and Load File

Operations: Load and Install GlobalPlatform's APDU commands.

FDP_ACF.1/ GemActivate_DAP Security attribute based access control

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

Subjects:

S.INSTALLER, defined in [PP-JCS] and represented by the GlobalPlatform Environment (OPEN) on the card, the Card Life Cycle attributes (defined in Section 5.1.1 of [GP]);

S.GEMACTIVATE, responsible to compute DAP on load package using GemActivate key and to compare computed DAP with received DAP to authorize loading and linking with restricted packages.

S.SD receiving the Card Content Management commands (through Load and Install APDUs)

Objects:

The GemActivate DAP Block, in case of application loading, referencing restricted packages (in particular the DESFIRE API);

The Load File or Executable File, in case of application loading, installation with a set of intended privileges and its targeted associated SD AID.

the following security attributes:

The CardState attribute, is the current state in the life cycle of the card, which may be either OP_READY, INITIALIZED, SECURED, CARD_LOCKED, or TERMINATED.

The Restricted Linking attribute, is a flag authorizing link with restricted pakage which may be either AUTHORIZED or BLOCKED.

The Registered Applications attribute specifies the Executable Files and application instances that have been installed on the card so far and their dependencies.

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

Runtime behavior rules defined by GlobalPlatform for:

loading (Section 9.3.5 of [GP]);

installation (Section 9.3.6 of [GP]);

where DAP verification is done using Gemactivate key by **Gemactivate Administrator**.

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

Rule GA-1: A loading and linking request for a package referencing restricted packages (in particular the DESFIRE API) may be accepted only if the APDU command specifying the request contains a DAP well-formed according to [GP22] and its verification using Gemactivate key by Gemactivate Administrator is successful.

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

Rule GA-2: When a loading and linking request for a package referencing restricted packages (in particular the DESFIRE API) fails, package is not installed and associated NVM is recovered.

Rule GA-3: When at least one of the rules for loading defined by GlobalPlatform [GP22] does not hold.

FMT_MSA.3/GemActivate_DAP Static attribute initialisation

FMT_MSA.3.1/GemActivate_DAP The TSF shall enforce the **GemActivate Security Domain access control policy** (see: application note) to provide **restrictive** default values for security attributes that are used to enforce the SFP.

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

Application note:

The **Restricted Linking** attribute for loading package with restricted import is set to Blocked by default. After a DAP verification, it is set to Authorized for the current loading and reset after successful or unsuccessful loading.

Keyset and Key used for DAP verification are under control of GemActivate and key is imported securely in Personalization phase.

7.1.4 DESFire EV1 Software

The following policy and security functional requirements can only be provided by the TOE if the DESFire EV1 Software is called by the Security IC Embedded Software. The subjects and objects described in the following policy are dedicated for the DESFire Emulation.

DESFire Access Control Policy The Security Function Policy (SFP) DESFire Access Control Policy uses the following definitions: The subjects are

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:

-- The DESFire card level data itself.

The DESFire EV1 Software 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 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 DESFire card level, 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 DESFire card level, applications, values and data files. There is a set of attributes for the DESFire card level, 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 DESFire card level, 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 authorized by cryptographic keys. These keys are considered as authentication data and not as security attributes. The DESFire card level 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).

FDP_ACC.1/DESFire Subset access control

FDP_ACC.1.1/DESFire The TSF shall enforce the **DESFire Access Control Policy** on **all subjects, objects, operations and attributes defined by the DESFire Access Control Policy**.

Application note:

The DESFire Access Control Policy is related to the data maintained by the DESFire EV1 Software.

FDP_ACF.1/DESFire Security attribute based access control

FDP_ACF.1.1/DESFire The TSF shall enforce the **DESFire Access Control Policy** to objects based on the following: **all subjects and objects and attributes associated to DESFire Access Control Policy**.

FDP_ACF.1.2/DESFire 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 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/DESFire The TSF shall explicitly authorise access of subjects to objects based on the following additional rules:

The Application Manager of an application can delete this application if this is allowed by a specific card attribute.

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

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.

FCS_COP.1/DESFire_TDES Cryptographic operation

FCS_COP.1.1/DESFire_TDES The TSF shall perform **encryption and decryption** in accordance with a specified cryptographic algorithm **Triple Data Encryption Algorithm (TDEA)** and cryptographic key sizes **of 112 or 168 bit** that meet the following: **list of standards: FIPS PUB 46-3 FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION DATA ENCRYPTION STANDARD (DES) Reaffirmed 1999 October 25, keying options 1 and 2**.

FCS_COP.1/DESFire_AES Cryptographic operation

FCS_COP.1.1/DESFire_AES The TSF shall perform **encryption and decryption** in accordance with a specified cryptographic algorithm **Advanced Encryption Standard (AES) algorithm** and cryptographic key sizes **of 128, 192 or 256 bit** that meet the following: **list of standards: FIPS PUB 197 FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, ADVANCED ENCRYPTION STANDARD (AES), National Institute of Standards and Technology, 2001 November 26**.

FMT_MSA.3/DESFire Static attribute initialisation

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

FMT_MSA.3.2/DESFire The TSF shall allow the **no subject** 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/DESFire Management of security attributes

FMT_MSA.1.1/DESFire The TSF shall enforce the **DESFire Access Control Policy** to restrict the ability to **freeze and modify** the security attributes **card attributes, application attributes and file attributes** to **the Administrator, Application Manager and Application User, or Everybody based on the refinement below.**

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..

Application note:

As an implication of the last rule, any subject that receives the modify abilities from the Application Manger 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/DESFire Specification of Management Functions

FMT_SMF.1.1/DESFire The TSF shall be capable of performing the following management functions: **Authenticate a user, Invalidating the current authentication state based on the functions: Selecting an application or the DESFire card level, 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.**.

FMT_SMR.1/DESFire Security roles

FMT_SMR.1.1/DESFire The TSF shall maintain the roles **Administrator, Application Manager, Application User, and Everybody**.

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

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

FDP_ITC.2.1/DESFire The TSF shall enforce the **DESFire Access Control Policy** when importing user data, controlled under the SFP, from outside of the TOE.

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

FDP_ITC.2.3/DESFire 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/DESFire 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/DESFire The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: **no additional rules**.

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

FPT_TDC.1.1/DESFire 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/DESFire The TSF shall use **data files or values can only be modified by their dedicated type-specific operations honoring the type-specific boundaries** when interpreting the TSF data from another trusted IT product.

Application note:

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.

FIA_UID.2/DESFire User identification before any action

FIA_UID.2.1/DESFire The TSF shall require each user to be successfully identified 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/DESFire User authentication before any action

FIA_UAU.2.1/DESFire 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/DESFire Multiple authentication mechanisms

FIA_UAU.5.1/DESFire The TSF shall provide **"none" and cryptographic authentication** to support user authentication.

FIA_UAU.5.2/DESFire The TSF shall authenticate any user's claimed identity according to the

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/DESFire Management of TSF data

FMT_MTD.1.1/DESFire The TSF shall restrict the ability to **change_default, modify and freeze** the **card master key, application master keys and application keys** to **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 as 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 the application assigned to him. 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/DESFire Trusted path

FTP_TRP.1.1/DESFire 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 and disclosure**.

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

FTP_TRP.1.3/DESFire The TSF shall require the use of the trusted path for **authentication requests with DES or AES, confidentiality and/or data integrity verification for data transfers protected with AES and based on a setting in the file attributes**.

FCS_CKM.4/DESFire Cryptographic key destruction

FCS_CKM.4.1/DESFire The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method **overwriting of memory** that meets the following: **none**.

FDP_ROL.1/DESFire Basic rollback

FDP_ROL.1.1/DESFire The TSF shall enforce **DESFire 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/DESFire 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/DESFire Replay detection

FPT_RPL.1.1/DESFire The TSF shall detect replay for the following entities: **authentication requests with DES or 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/DESFire The TSF shall perform **rejection of the request** when replay is detected.

FDP_ACC.1/DESFire_API Subset access control

FDP_ACC.1.1/DESFire_API The TSF shall enforce the **DESFire_API Access Control Policy** on **all subjects, objects, operations and attributes defined by the DESFire_API Access Control Policy**.

Application note:

The DESFire_API Access Control Policy is related to the access to methods defined in the interface of DESFire API accessing to the DESFire EV1 data. Such access is restricted to the authorized applets using registration of CODEC AID.

FDP_ACF.1/DESFire_API Security attribute based access control

FDP_ACF.1.1/DESFire_API The TSF shall enforce the **DESFire Access Control Policy** to objects based on the following: **all subjects and objects and attributes associated to DESFire Access Control Policy**.

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

The DESFIRE_API administrator can register and unregister applications using CODEC AID stored in DESFIRE EV1 data during personalization phase.

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

Only registered applications can access to any methods defined in the interface of the DESFIRE_API by supplying CODEC AID to be identified.

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

Nobody can access to DESFIRE shareable interface prior to be identified by supplying CODEC AID already registered during personalization phase.

FIA_UID.2/DESFire_API User identification before any action

FIA_UID.2.1/DESFire_API The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

Application note:

Identification of a user (an application) is performed upon a request based on CODEC AID already registered by DESFIRE EV1 software. Before any identification request is issued the application is identified as unknown.

FMT_SMR.1/DESFire_API Security roles

FMT_SMR.1.1/DESFire_API The TSF shall maintain the roles **[DESFIRE_API Administrator]**.

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

FMT_MSA.1/DESFire_API Management of security attributes

FMT_MSA.1.1/DESFire_API The TSF shall enforce the **DESFIRE_API access control SFP** to restrict the ability to **modify** the security attributes **state (deactivated, activated) allowing use of DESFIRE_API services to authorized application**s to **DESFIRE_API Administrator under control of Gemalto**.

FMT_SMF.1/DESFire_API Specification of Management Functions

FMT_SMF.1.1/DESFire_API The TSF shall be capable of performing the following management functions: **access to services delivered by DESFIRE API**.

FMT_MSA.3/DESFire_API Static attribute initialisation

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

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

FDP_ETC.1/DESFire_API Export of user data without attributes

FDP_ETC.1.1/DESFire_API The TSF shall enforce the **DESFire_API Access Control Policy** when exporting user data, controlled under the SFP, outside of the TOE.

FDP_ETC.1.2/DESFire_API The TSF shall export the user data without the user data's associated security attributes

Application note: Export of DESFIRE EV1 data to authorized applications is only possible through DESFIRE_API interface according to DESFIRE_API access control policy.

FDP_ITC.1/DESFire_API Import of user data without attributes

FDP_ITC.1.1/DESFire_API The TSF shall enforce the **DESFire_API Access Control Policy** when importing user data, controlled under the SFP, from outside of the TOE.

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

FDP_ITC.1.3/DESFire_API The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: [assignment: access to DESFIRE_API Interface contains methods with restricted access defined in *DESFIRE_API access control policy*].

Application note: Import of DESFIRE EV1 data from authorized applications is only possible through DESFIRE_API interface according to DESFIRE_API access control policy.

7.2 Security Assurance Requirements

The security assurance requirement level is EAL4 augmented with ALC_DVS.2 and AVA_VAN.5.

7.3 Security Requirements Rationale

Chapter content has been removed in Public version.

8 TOE Summary Specification

8.1 TOE Summary Specification

8.1.1 Basic TOE

8.1.1.1 GP

GP.CardContentManagement

This security function provides the capability and a dedicated flow control for the loading, installation, extradition, registry update, selection and removal of card content and especially executable files and application instances.

Such features are offered to the Card Issuer and its business partners, allowing the Card Issuer to delegate card content management to an Application Provider according to privileges assigned to the various security domains on the card.

It supports delegated management and it can use DAP or Mandated DAP verification and generation of Reception token.

It also checks that only the card management commands specified and allowed at each state of the smart card's life cycle are accepted, and ill-formed ones are rejected with an appropriate error response.

GP.SecurityDomain

This security function provides security domain management: as SD creation, SD selection, SD privileges setting, SD deletion in SD hierarchy. It provides means to associate or extradite an application to a security domain in order to provide services (as secure channel) to the dedicated application without sharing the related keys stored in SD. It also provides Keyset Management in SD, with Key Set creation, Key set deletion, key importation, replacement, or deletion in Key Set.

Security Domains are privileged Applications as defined in [GP2.2.2 § 7], holding cryptographic keys to be used to support Secure Channel Protocol operations and/or to authorize card content management functions. There are different types of security domain with dedicated privileges and associated operations: ISD Security domain, Supplementary Security domains, and Controlling Authority Security domains.

ISD Security domain as defined in [GP2.2.2 §7.1], is the mandatory Security Domain, implicitly selected if the Application implicitly selectable on the same logical channel of the same card I/O interface is removed. It inherits of the Final Application privilege if the Application with that privilege is removed.

Supplementary Security Domains are privileged Applications with dedicated privileges:

Token Verification Privilege as described in [GP22 §9.1.3.1]

Delegated Management Privilege as described in [GP22 §9.1.3.3]

Global Delete Privilege as described in [GP22 §9.1.3.4]

Global Lock Privilege as described in [GP22 §9.1.3.5]

Receipt Generation Privilege as described in [GP22 §9.1.3.6]

Controlling Authority Security Domain is a supplementary Security Domain dedicated to the Controlling Authority with dedicated privileges. It contains Security Domains cryptographic keys needed to confidentially personalize an initial set of Secure Channel Keys of an APSD.

GP.SCP

This security function manages Secure Channel protocol according to [GP22] annex D,E and [GP22-A] and [TS 102.225].

GP.ISD

This security function manages the Issuer Security Domain with associated functions and dedicated privileges as defined in [GP2.2.2 §7.1].

GP.SecureChannel

This security function provides a secure communication channel between a card and an off-card entity during an Application Session. It provides an APDU flow control using the Command security level check according to Card Life cycle and type of APDU.

A Secure Channel Session is divided into three sequential phases:

- Secure Channel Initiation when the on-card Application and the off-card entity have exchanged sufficient information enabling them to perform the required cryptographic functions. The Secure Channel Session initiation always includes (at least) the authentication of the off-card entity by the on-card Application; performing also the setting of the Command security level used for the session.
- Secure Channel Operation when the on-card Application and the off-card entity exchange data within the cryptographic protection of the Secure Channel Session. The Secure Channel services offered may vary from one Secure Channel Protocol to the other;
- Secure Channel Termination when either the on-card Application or the off-card entity determines that no further communication is required or allowed via an established Secure Channel Session.

The following services are provided by the Secure Channel as defined in [GP section 4.3.2 and §10- Secure Communication] using SCP 01 or SCP 02 or SCP 80.

- Entity authentication in which the card or the off-card entity proves its authenticity to the other entity through a cryptographic exchange, based on session key generation and a dedicated flow control; For SCP80, envelope APDU shall contain secured packet structure defined in [ETSI 102.225 §5] and Anti-replay mechanism is proposed optionally using a counter defined in [ETSI 102.225 §5.1.4]
- Integrity and authentication in which the receiving entity (the card or off-card entity) ensures that the data being received from the sending entity (respectively the off-card entity or card) actually came from an authenticated entity in the correct sequence and has not been altered;
- Confidentiality in which data being transmitted from the sending entity (the off-card entity or card) to the receiving entity (respectively the card or off-card entity) is not viewable by an unauthenticated entity.

GP.GPRegistry

This security function provides accesses to the GlobalPlatform Registry used for:

Store card management information;

Store relevant application management information (e.g., AID, associated Security Domain and Privileges);

Support card resource management data;

Store Application Life Cycle information;

Store card Life Cycle information;

Track any counters associated with logs.

The contents of the GlobalPlatform Registry may be accessed by administrative commands or by applet using a dedicated GP_API.

GP.SSD

This security function manages supplementary Security Domains with associated functions and dedicated privileges as defined in [GP22 §9.1].

Application note:

Token Verification Privilege as described in [GP22 §9.1.3.1] Authorized Management Privilege as described in [GP22 §9.1.3.2] Delegated Management Privilege as described in [GP22 §9.1.3.3] Global Delete Privilege as described in [GP22 §9.1.3.4] Global Lock Privilege as described in [GP22 §9.1.3.5] Receipt Generation Privilege as described in [GP22 §9.1.3.6]

8.1.1.2 JCS

This section defines the security functions to be achieved by the JCS part of the TOE.

JCS.APDUBuffer

The security function maintains a byte array buffer accessible from any applet context. This buffer is used to transfer incoming APDU header and data bytes as well as outgoing data according to [JCAPI301]. The APDU class API is designed to be transport protocol independent T=0, T=1, T=CL (as defined in ISO 7816-3).

Application note:

ADPU buffer is a JCRE temporary entry point object where no associated reference can be stored in a variable or an array component.

JCS.ByteCodeExecution

This security function realizes applet bytecode execution according to JCVM rules [JCVM].

The JCVM execution may be summarized in JCVM interpreter start-up, bytecode execution and JCVM interpreter loop. The applet bytecode execution consists in:

fetching the next bytecode to execute according to the applet's code flow control,

decoding the next bycode,

executing the fetched bytecode.

The JCVM manages 5 types of objects: persistent objects, transient objects, persistent arrays (boolean, byte, short, int or reference), transient arrays (boolean, byte, short, int or reference) and static field images. For each type of object, different types of control are performed [see JCVM §4].

JCS.Crypto

The security function offers the following services to applets thanks to JavaCard API:

Generation of random number as defined in [JCAPI301] and conformant to ANSSI standard to be used for key values or challenges during external exchanges,

Computation of checksum CRC16 and CRC32 conformant with ISO3309,

Ciphering and deciphering operation using DES algorithm in ECB and CBC mode with padding scheme (NOPAD, ISO9797 or PKCS #5),

Ciphering and deciphering operation using AES (128 bits) algorithm in ECB and CBC mode with padding scheme (NOPAD),

Ciphering and deciphering operation using RSA with CRT algorithm in mode ISO14888, with padding scheme (ISO9796 or PKCS #1),

Data Hash operation for message digests using SHA-2 algorithm (SHA-256, SHA-384, SHA-224),

Generation of an signature of a byte array, and verifying an signature stored in a byte array using a generation of 20-byte SHA-2 message digest using RSA algorithm with PKCS#1-PSS padding scheme,

Generation of 4-byte or 8-byte MAC using DES (112 or 168 bits key) algorithm according to ISO9797-1,

Generation of 16-byte MAC using AES algorithm in CBC mode (MAC_128) with padding scheme (NOPAD).

These operations are performed in a way to avoid revealing the key values. If the applet specifies an algorithm that the platform does not support, the JCRE refuses to perform the cryptographic operation and generates an exception. Even if [JCAPI301] specifies some other algorithms or parameters for cryptographic operations, the use of these other values are not advised; and clearly out of scope of the TOE. See [USR] for details.

JCS.EraseResidualData

The security function ensures that sensitive data are locked upon the following operations as defined in [JCRE301]:

Deletion of package and/or applications,

Deletion of objects.

There are erased when space needs to be reused for allocation of new object.

This security function also ensures that the sensitive temporary buffers (transient object, bArray object, APDU buffer, Cryptographic buffer) are securely cleared after their usage with respect to their life-cycle and interface as defined in [JCRE301], transient object at reset or allocation and persistent object are erased at allocation for new object.

JCS.Exception

This security function manages throwing of an instance of Exception class in the following cases:

a SecurityException when an illegal access to an object is detected,

a SystemException with an error code describing the error condition,

a CryptoException in case of algorithm error or illegal use,

any exception decided by the applet or the JCRE handled as temporary JCRE entry point object with associated JC API. It also offers a means to applet to handle exception and to JCRE to handle uncaught exception by applets.

JCS.Firewall

This security function enforces the Firewall access control policy and the JCVM information flow control policy at runtime. It defines how accessing the following items: Static Class Fields, Array Objects, Class Instance Object Fields, Class Instance Object Methods, Standard Interface Methods, Shareable Interface Methods, Classes, Standard Interfaces, Shareable Interfaces, Array Object Methods.

Based on security attributes [Sharing, Context, Lifetime], it performs access control to object fields between objects and thows security exception when access is denied. Thus, it enforces applet isolation located in different packages and controls the access to global data containers shared by all applet instances.

The JCRE shall allocate and manage a context for each Java API package containing applets. The JCRE maintains its own context a special system privileges so that it can perform operations that are denied to contexts of applets.

JCS.KeyManagement

This security function enforces key management for the different associated operations: key building, key agreement, key generation, key importation, key exportation, key masking, key destruction using standard API defined in [JCAPI301].

Key generation support generation of RSA key pairs using a secure random number generator compliant with ANSSI's Standard security level for cryptography operations.

Key agreement enables an applet to agree on a shared secret with the external, with a method conformant to [JCAPI301]. It is built to avoid disclosure of this secret to third parties observing exchange done for key agreement.

Key masking protects the confidentiality of cryptographic keys from being read out from the memory. It ensures the service of accessing and modifying them.

Key destruction disables the use of a key both logically and physically. Reuse is only possible after erase.

Key importation and exportation is done using method protecting confidentiality and integrity of key.

JCS.OutOfLifeDataUndisclosure

This security function ensures that sensitive data are locked until postponed erasure on the following operations:

Deletion of persistent and transient objects according to [JCRE301].

JCS.OwnerPIN

This security function supplies to applet a mean to assume a user identification and authentication with the OwnerPin class conformant to [JCAPI301].

It offers to create a PIN and store it securely in the persistent memory. It allow access to PIN value only to perform a secure comparison between a PIN stored in the persistent memory and a data received as parameter.

A method returns a positive result if a valid Pin has been presented during current session. If the PIN is not blocked and the comparison is successful, the validated flag is set to and the try counter is set to its maximum, otherwise the authentication fails and the associated try counter is decremented. When the validated flas is set, it is assumed that the user is authenticated.

When the try counter reaches zero, the PIN is blocked and the authentication is no more possible until the PIN is unblocked.

JCS.Package

This security function manages packages. Package is a structural item defined for naming, loading, storing, execution context definition. There are rules for identification of package, for structure check and access rules definition. If inconsistent items are found during checks, an error message is sent.

JCS.RNG

This security function provides random value using a given algorithm with or without a seed as defined in [JCAPI301].

JCS.RunTimeExecution

This security function provides a secure run time environment and deals with:

Instance registration or deletion,

Application selection,

Applet opcode execution,

JCS API methods execution,

Logical channel management,

APDU flow control, dispatch and buffer management,

JCRE memory and context management,

JCRE reference deletion,

JCRE access rights,

JCRE throw exception,

JCRE security reaction.

8.1.1.3 SecureAPI

SA.FlowControl

The security function provides means to application to control execution flow, to detect any failure and to react if required.

SA.SecureOperation

The security function provides means to application to execute securily data transfer and comparison, to detect any failure during operation and to react if required.

SA.RandomDelay

The security function provides means to introduce dummy operations leading to unobservability of sensitive operation.

8.1.1.4 GemActivate

GA.OptionalServiceActivation

Activation is only possible for deactivated services defined in registry. Activation is done by changing internal state of optional platform service: cryptographic algorithm, package, applet instance, scalability (extension of available NVM) and NFC interface (SWP, HCI

gate). The command is available only for GemActivate under control of GemActivate Administrator. GemActivate is accessible only using a secure channel under control of MNO.

GA.ServiceAudit

The application allow MNO or GemActivate administrator audit of actual state (deactivated, activated, inhibited)of each optional platform service described in platform registry.

GA.GemActivateActivation

The application can be activated by GemActivate Administrator only if the following conditions are fulfilled:

if activation command is consistent,

if ratification counter limit is not reached,

if anti replay verification has not failed,

if activation signature verification has not failed.

The application allows activation of the following optionnal services: Standard or private cryptographic algorithms, optional packages, unactivated applets loaded in Pre-issuance, optional NFC gate, available user memory size in NVM.

GA.GemActivateDAPVerification

During loading and linking operations for a package importing a restricted package, a check that a DAP has been verified with GemActivate is performed prior to authorize such operation.

8.1.1.5 DESFire

SS.DF_TRANS: DESFire Transaction Protection

The transaction mechanism implemented by SS.DF_TRANS ensures that either all or none of the (modifying) commands within a transaction are performed. The transaction mechanism is active for backup data files, value files, linear record files and cyclic record files, it is not active for standard data files. All file types with the exception of "standard data files" are called "backup files" in the following. SS.DF_TRANS is always active for the respective file types. This means that for every modifying operation with a backup file an explicit commit request must be issued in order to let the modifications take effect. Several reasons will abort a transaction: These are the explicit abort request, chip reset, an authentication request, a "select" command or any failure of a command.

SS.DF_TYPECHECK: DESFire Filetype Consistency Check

SS.DF_TYPECHECK ensures the type consistency of the file types stored by the TOE. For value files the check comprises over- or underflow. Furthermore size limitations of files are obeyed and SS.DF_TYPECHECK ensures that records read/writes are handled specific to the type of the record file.

SS.DF_AUTH: DESFire Authentication

The TOE provides an authentication mechanism to separate authorized subjects from unauthorized subjects. The authentication of subjects is performed by a cryptographic

challenge-response. The TOE supports the cryptographic algorithms 2-key Triple-DES, 3-key Triple-DES and 128-bit AES; for DES according to FIPS PUB 46-3 [19] and for AES according to FIPS PUB 197 [20]. The authentication mechanisms are implemented using Platform JavaCard API based on the cryptographic coprocessors and the hardware random number generator provided by the hardware platform. The authentication mechanisms are protected against attacks like e.g. replay. SS.DF_AUTH identifies the user to be authenticated by the currently selected context (specific application, chosen by a "select" command) and the key number indicated in the authentication request. By default and before any authentication request SS.DF_AUTH identifies and authenticates the role Everybody. The roles Administrator, Application Manager, Application User are authenticated during the authentication request by the knowledge of the respective cryptographic key. The authentication state is remembered by SS.DF_AUTH and the authentication need not to be performed again as long as none of the following events occur: Issue of a "select" command, occurrence of any error during the processing of commands, change of the key that was used for authentication and reset (any cause, either internal or external reset). These events will reset the authentication state to the default (Everybody). Additionally, if the Application Manager deletes his application the authentication state will be reset as an implication.

SS.DF_CONFID: DESFire Communication Confidentiality

The TSF SS.DF_CONFID provides a mechanism to protect the communication against eavesdropping. In order to do this the communication can be encrypted. The encryption is performed based on the option in the file attributes. There options can be changed by the file owner (i.e. the subject that has the right to "change attribute" for a file). The encryption algorithm is the same as the one used during authentication for the session, however SS.DF_CONFID only supports the AES algorithm, therefore it is bound to authentications with this algorithm. Note that the TSF SS.DF_CONFID can be activated after authentication performed with SS.DF_AUTH. SS.DF_CONFID can also be configured to add data to the unencrypted communication stream that enables the terminal to detect integrity violations, replay attacks or man-in-the-middle attacks. If an encrypted communication is requested, SS.DF_CONFID also verifies the data sent by the terminal and returns an error code if such an attack is detected. The detection mechanism covers all frames exchanged between the terminal and the DESFire EV1 Software up to the current encrypted frame. Therefore SS.DF_CONFID can detect any injected/modified frame in the communication before the transfer of the encrypted frame, but it cannot detect what frame was injected/modified.

SS.DF_ACCESS: Access Control to DESFire Data

SS.DF_ACCESS provides an access control mechanism to the objects and security attributes of the DESFire EV1 Software. The access control mechanism assigns subjects - (possibly multiple) Application Users - to 4 different groups of operations on files. For data files, the operations are "read", "write", "read and write" and "change attribute". For values the operations are "read and decrease", "read, decrease, limited increase", "read, decrease, limited increase, increase" and "change attribute". One subject can be assigned to each group of file operations. The special subjects "Everybody" and "Nobody" can also be assigned. For applications of the DESFire EV1 Software the operations are "create file" and "delete file". These operations can be assigned to the Application Manager or to everybody. The SS.DF_ACCESS provides an access control mechanism to the objects and security attributes of the DESFire EV1 Software. The access control mechanism assigns

subjects - (possibly multiple) Application Users - to 4 different groups of operations on files. For data files, the operations are "read", "write", "read and write" and "change attribute". For values the operations are "read and decrease", "read, decrease, limited increase", "read, decrease, limited increase, increase" and "change attribute". One subject can be assigned to each group of file operations. The special subjects "Everybody" and "Nobody" can also be assigned. For applications of the DESFire EV1 Software the operations are "create file" and "delete file". These operations can be assigned to the Application Manager or to everybody. The assignment is stored in the application attributes. If a file is created the file attributes must be supplied with the create request. On the DESFire card level the operations are "create application" and "delete application". These operations can be assigned to the Administrator or to Everybody. The assignment is stored in the card attributes. If an application is created the application attributes must be supplied with the create request. A "delete application" operation will securely delete all application keys by overwriting them with random values.

SS.DF_ACCESS also controls access to the security attributes and the authentication data. The card attributes and the card master key can only be changed by the Administrator, as long as the Administrator does not freeze the card attributes or freezes the card master key. The application attributes and application master keys can be changed by the Application Manager, as long as the Application Manager does not freeze the application attributes or the application master key. Additionally the Application Manager can change the Application User keys and decide if the Application Users can change their keys or not. For files, the attributes can be changed by the subject that has the "change attribute" right. SS.DF_ACCESS allows the Administrator to specify a default application master key and application keys that will be used when an application is created.

SS.DF_CRYPTO: DESFire Cryptographic operations

The application performs cryptographic operations using JCS API and dedicated API in specific cases.

SS.DF_API_ACCESS Access Control to DESFire Data through DESFIRE API

SS.DF_API_ACCESS provides an access control mechanism to the objects and security attributes of the DESFire EV1 Software. The DESFIRE API defines an interface with a set of methods to access to DESFire EV1 data. The access control mechanism to such method is built on identification of application requesting access. Two levels of control are defined, access to a dedicated DESFire EV1 shareable interface and then individual access to any methods of the interface. For each operation, application identification is required using CODEC AID defined during Personnalization phase. List of authorized CODEC AID are initialized and stored in DESFire EV1 Software to perform CODEC AID comparison for identification.

8.1.1.6 OS

This section defines the security functions to be achieved by the OS part of the TOE.

OS.Atomicity

The security function performs write operations atomically on complex type or object in order to avoid incomplete update. Prior to be written, the data are stored in an atomic back-up area. In case on writing interrupt, the only two possible values are: initial value if

writing is not started or final value if writing is started. At next start-up, the atomic back-up area is check to finalize interrupted writing.

OS.Memory Management

The security function allocates memory areas and performs access control to memory areas to avoid unauthorized access. It manages circular writing to avoid instable memory state. It assumes memory recovery in case of error detection. It offers (when required) confidentiality services for data storage: Ciphering / Deciphering of Data in RAM or in FLASH, Scrambling / Unscrambling of Data in RAM or in FLASH.

8.1.1.7 IC

IC.Limited FaultTolerance

The TSF manages a certain number of faults or errors that may happen, related to memory content, CPU, Random generation and cryptographic operation, thus preventing risk of malfunction. It is related to FRU_FLT.2 from [ST/IC].

IC.Secure State

The TSF provides preservation of secure state managing security violation resulting in an immediate reset. It is related to FPT_FLS.1 from [ST/IC].

IC.LIM.Capability (TEST)

The TSF ensures that test capability is unavailable in USER configuration. It is related to FMT_LIM.1 [TEST] from [ST/IC].

IC.LIM.Capability (ISSUER)

The TSF ensures that secure flash loader and test capability are unavailable in USER configuration. It is related to FMT_LIM.1 [ISSUER] from [ST/IC].

IC.ModeControl

The TSF ensures that only defined modes are available: TEST, ISSUER, USER configuration. It is related to FMT_LIM.2 [TEST] & ISSUER] from [ST/IC].

IC.Audit Storage

The TSF provides command to store data for audit purpose using commands only available to authorized process. It is related to FAU_SAS.1 from [ST/IC]. IC_Audit_Storage is used for TOE identification in phase 3 & 4.

IC.Resistance to Physical Attack

The TSF ensures resistance to physical tampering using features against probing and an active shield detecting integrity violation. It is related to FPT_PHP.3 from [ST/IC].

IC.Internal Data Transfer Protection

The TSF prevents disclosure of internal and user data thanks to memory scrambling and encryption, bus encryption... It is related to FDP_ITT.1, FPT_ITT.1, FDP_IFC.1 from [ST/IC].

IC.Random Number Generation

The TSF produces AIS31-qualified random numbers that can be directly used in embedded software. It is related to FCS_RNG.1 from [ST/IC].

IC.Cryptoaccelerator

The TSF provides EDES accelerator to perform DES and TDES encryption and decryption conformant to FIPS PUB 46-3.

The TSF provides arithmetic primitives to be used in more complex computation in software cryptographic library.

It is related to FCS_COP.1 from [ST/IC].

It also uses RNG, arithmethic primitives of Nescrypt. But there is no usage of NesLib.

IC.Memory Protection

The TSF enforces a default memory protection policy when none other is programmed by the embedded software. It is related to FMT_MSA.3 from [ST/IC].

IC.MPU

The TSF provides a dynamic Memory protection unit (MPU) that can be configured by the ES. It is related to FMT_MSA.1, FMT_SMF.1 from [ST/IC].

IC.Loading Access Control

The TSF provides an access control to loading. The Standard Loader instructions and/or Advanced Loader instructions can be executed only if valid passwords have been presented. It is related to FDP_ACC.2, FDP_ACF.1 from [ST/IC].

8.2 SFRs and TSS

Chapter content has been removed in Public version.

9 References, Glossary and Abbreviations

9.1 External References

Reference	Title
[CC-1]	Common Criteria for Information Technology Security Evaluation Part 1: Introduction and general model CCMB-2012-09-001, version 3.1 Release 4, September 2012.
[CC-2]	Common Criteria for Information Technology Security Evaluation Part 2: Security Functional Requirements CCMB-2012-09-002, version 3.1 Release 4, September 2012.
[CC-3]	Common Criteria for Information Technology security Evaluation Part 3: Security Assurance Requirements CCMB-2012-09-003, version 3.1 Release 4, September 2012.
[CEM]	Common Methodology for Information Technology Security Evaluation CCIMB-2012-09-004, version 3.1 Release 4, September 2012.
[Comp]	CCDB, Composite product evaluation for Smart Cards and similar devices, September 2012, Version 1.2, April 2012, CCDB-2012-04-001
[DCSSI2741]	Règles et recommandations concernant le choix et le dimensionnement des mécanismes cryptographiques de niveau de robustesse standard N° 2741/SGDN/DCSSI/SDS/LCR Version 1.10
[FIPS 46-3]	FIPS 46-3: DES Data Encryption Standard (DES and TDES). National Institute of Standards and Technology
[FIPS 197]	FIPS 197: AES Advanced Encryption Standard. National Institute of Standards and Technology.
[FIPS 180-2]	FIPS-46-3: Secure Hash Standard (SHA). National Institute of Standards and Technology.
[GP22]	Global Platform Card Specification 2.2.1
[GP-CCCM]	GlobalPlatform, Card Confidential Card Content Management, Card specification v2.2 – Amendment A,V1.0
[GP-RAM]	GlobalPlatform, Remote Application Management over HTTP Card Specification v 2.2 - Amendment B, Version 1.1
[GP-CCS]	GlobalPlatform, Card Contactless Services, Card specification v2.2 – Amendment C, V1.0
[GP-SCP]	GlobalPlatform, Secure Channel Protocol 03 Card Specification v 2.2 – Amendment D Version 1.1
[GP-UICC]	GlobalPlatform Card UICC Configuration Version 1.0.1
[IC_CERTIF]	ST33 CC certificates : ANSSI-CC-2011/07 and maintained with ANSSI-CC-2011/07-M01 and ANSSI-CC-2011/07-M02
[ISO 7816-4]	Identification cards - Integrated circuit(s) cards with contacts, Part 4: Interindustry commands for interchange.
[ISO 7816-6]	Identification cards - Integrated circuit(s) cards with contacts, Part 6: Interindustry data elements.
[ISO 7816-9]	Identification cards - Integrated circuit(s) cards with contacts, Part 9: Additional Inter industry commands and security attributes.
[ISO 9796-2]	ISO/IEC 9796-2 Information technology -- Security techniques -- Digital signature schemes giving message recovery -- Part 2: Integer factorization based mechanisms

Reference	Title
[JCRE301]	Runtime Environment Specification Java Card™ Platform, Version 3.0.1, Classic Edition, May 2009
[JCVM301]	Virtual Machine Specification Java Card™ Platform, Version 3.0.1, Classic Edition, May 2009
[JCAPI301]	Java Card™ APIs specification, Version 3.0.1, Classic Edition, May 2009
[JAVASPEC]	The Java Language Specification. Third Edition, May 2005. Gosling, Joy, Steele and Bracha. ISBN 0-321-24678-0
[JIL]	Joint Interpretation Library Composite product evaluation for Smart Cards and similar devices Version 1.0 September 2007
[NXP-DESFIRE]	MIFARE DESFire Functional Specification, NXP Semiconductors, Business Line Identification
[PP-BSI-0035]	Security IC Platform Protection Profile Version 1.0 15.06.2007
[PP-JCS]	Java Card™ System Protection Profile “Open Configuration” Version 3.0, May 2012
[PP-USIM]	(U)SIM Java Card Platform Protection Profile Basic Configuration V2.0.2, June 2010
[RFC2085]	HMAC-MD5 IP Authentication with Replay Prevention
[RFC2104]	HMAC: Keyed-Hashing for Message Authentication
[RSA PKCS#1]	PKCS #1 v2.1: RSA Cryptography Standard
[ST/IC]	Security Target of ST33F1M SMD_ST33F1M_ST_10_001 Rev 01.00
[STM_SG]	ST33F1M Security Guidance
[TS03.19]	ETSI 3GPP TS 03.19, Subscriber Identity Module Application Programming Interface (SIM API) for Java Card™; Stage 2
[TS 51.011]	ETSI 3GPP TS 51.011, 3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; Specification of the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface (Release 4)
[TS 51.013]	ETSI 3GPP TS 51.013, 3rd Generation Partnership Project; Technical Specification Group Terminals; Test specification for SIM API for Java Card™ (Release 4)
[TS 51.014]	ETSI 3GPP 51.014, 3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; Specification of the SIM Application Toolkit for the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface (Release 4)
[TS 102.221]	ETSI 3GPP TS 102.221, Smart cards; UICC-Terminal interface; Physical and logical characteristics (Release 6)
[TS 102.222]	ETSI 3GPP TS 102.222, Integrated Circuit Cards (ICC); Administrative commands for telecommunications applications (Release 6) – v 6.0.0 (2003-02).
[TS 102.223]	ETSI 3GPP TS 102.223, Smart Cards; Card Application Toolkit (CAT) (Release 6)
[TS 102.224]	ETSI 3GPP TS 102.224, Smart cards; Security mechanisms for UICC based Applications - Functional requirements (Release 6)
[TS 102.225]	ETSI 3GPP TS 102.225, Smart cards; Secured packet structure for UICC based applications (Release 6) –
[TS 102.226]	ETSI 3GPP TS 102.226, Smart Cards; Remote APDU structure for UICC based applications (Release 6)
[TS 102.240]	ETSI 3GPP TS 102.240, Smart Cards; UICC Application Programming Interface and Loader Requirements; Service description; (Release 6)
[TS 102.241]	ETSI 3GPP TS 102.241, UICC API, 3GPP Release 6

Reference	Title
[TS 102.613]	ETSI 3GPP TS 102.613, UICC - Contactless Front-end (CLF) Interface; Part 1: Physical and data link layer characteristics (Release 7)
[TS131.111]	ETSI 3GPP TS 131.111, Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Universal Subscriber Identity Module (USIM) Application Toolkit (USAT) (Release 6)
[T131.130]	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)
[Standard APP]	D1186227_guidance_for_basic_application_development_on_Upteq_NFC_products.pdf

9.2 Internal References

Reference	Title
[ST_PLF]	Security Target : DESFIRE 1.1 on UpTeq NFC 2.1.3_Generic ST_ D1314435
[FSP_PLF]	Upteq NFC2.1 Platform Functional Specification D1145952
[TDS_PLF]	DESFire EV1 on NFC 2.1 TOE Design Specification D1319369
[ARC_PLF]	Upteq NFC2.1 Platform TOE Security Architecture D1145954
[IMP_PLF]	Upteq NFC2.1 Platform Implementation representation D1226485
[PRE_PLF]	Upteq NFC2.1 Platform Preparation Guidance D1310796
[OPE_PLF]	Upteq NFC 2.Y Platform Operational Guidance with Controlling Authority and optional Verification Authority D1310910
[COV_PLF]	Upteq NFC2.1 Platform Analysis of test coverage D1307421
[DPT_PLF]	Upteq NFC2.1 Platform Analysis of the depth of testing D1307422
[FUN_PLF]	Upteq NFC2.1 Platform Functional Test Documentation D1307420
[CMC_PLF] [CMS_PLF]	Upteq NFC2.1 Platform Configuration Management Plan D1292707 Upteq NFC2.1 Platform Configuration Management Scope D1292708
[LCD_PLF]	Upteq NFC2.1 Platform Life cycle Support D1311609
[DVS_PLF]	Upteq NFC2.1 Platform Development Security Documentation D1311618
[DEL_PLF]	Upteq NFC2.1 Platform Delivery Documentation D1311650
[TAT_PLF]	Upteq NFC2.1 Platform documentation of development tools D1311600
[COMP_PLF]	Upteq NFC2.1 Platform Composition with Hardware D1145972

9.3 ABBREVIATIONS

Abbreviation	Description
AES	Advanced Encryption Standard
AID	Applet Identifier
APDU	Application Protocol Data Unit
API	Application Programmer Interface
CBC	Cipher Block Chaining
CC	Common Criteria
CM	Card Manager
CPLC	Card Production Life Cycle
DAP	Data Authentication Pattern
DES	Data Encryption Standard
DS	Dedicated Software
EAL	Evaluation Assurance Level
GP	Global Platform
HMAC	Keyed-Hash Message Authentication Code
IC	Integrated Circuit
JCRE	Java Card Runtime Environment
JCS	Java Card System
JCVM	Java Card Virtual Machine
MAC	Message Authentication Code
OSP	Organizational Security Policy
PP	Protection Profile
RNG	Random Number Generation
RSA	Cryptographic module "Rivest, Shamir, Adleman"
SHA-2	Cryptographic module "Secure hash standard"
ST	Security Target
TOE	Target of Evaluation.
VA	Verification Authority

9.4 Glossary

Term	Definition
Application	Instance of an Executable Module after it has been installed and made selectable
APDU	Standard communication messaging protocol between a card accepting device and a smart card
Card Administrator	The card administrator is an external entity issuing the card and performing main functions of card administration (as Card life cycle Management). It is usually the Card Issuer or MNO.
Controlling Authority	A Controlling Authority is entity independent from the MNO represented on the (U)SIM card and responsible for securing the keys creation and personalization of the Supplementary Security Domains.
DAP Block	Part of the Load File used for ensuring Load File Data Block verification
DAP Verification	A mechanism used by a Security Domain to verify that a Load File Data Block is authentic
Delegated Management	Pre-authorized Card Content changes performed by an approved Application Provider
Executable Load File	Actual on-card container of one or more application's executable code. It may reside in Immutable Persistent Memory or may be created in Mutable Persistent Memory as the resulting image of a Load File Data Block.
Executable Module	Contains the on-card executable code of a single application present within an Executable Load File
Issuer Security Domain	The primary on-card entity providing support for the control, security, and communication requirements of the card administrator (the Card Issuer or MNO)
Load File	A file transferred to a GlobalPlatform card that contains a Load File Data Block and possibly one or more DAP Blocks
Load File Data Block	Part of the Load File that contains one or more application(s) or libraries and support information for the application(s) as required by the specific platform
Load File Data Block Hash	A value providing integrity for the Load File Data Block
Message Authentication Code	A symmetric cryptographic transformation of data that provides data origin authentication and data integrity
Secure Channel	A communication mechanism between an off-card entity and a card that provides a level of assurance, to one or both entities
Secure Channel Protocol	A secure communication protocol and set of security services
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)
Supplementary Security Domain	A Security Domain other than the Issuer Security Domain dedicated to Application provider.
Token	A cryptographic value provided by a Card Issuer as proof that a Delegated Management operation has been authorized
Verification Authority	The Verification Authority (VA), is a trusted third party, acting on behalf of the MNO and responsible for the verification of application and generation of digital evidence to be checked by the TOE during the loading process.