MultiAppV1.0 Platform

Security Target

MODIFICATION SHEET

Date	Modifications	Author
September 11, 2009	Creating from the evaluated Security Target	Quang-Huy Nguyen

TABLE OF CONTENTS

Figures

Tables

1 ST introduction (ASE_INT)

1.1 ST reference

Title: MultiAppV1.0 platform Security Target

Origin: GEMALTO

ITSEF: Stratsec

Evaluation scheme: Australasian Information Security Evaluation Program

Component	Reference/Version	Supplier
MultiAppV1.0 platform embedded in ROM	1.0	Gemalto
P5CD144 secure smart card controller including an **AIS31-certified Deterministic Random Number Generator** (DRNG)	V0B	NXP

Table 1. TOE components

1.2 TOE reference

TOE title: MultiAppV1.0 Platform

Product name: MultiApp ID v1.0 Citizen Combi 144K

Product reference: T1004246

Commercial name: MultiApp ID 144K

1.3 TOE overview

The Target of Evaluation (TOE) is composed of the **MultiAppV1.0** platform. The platform includes the hardware and the operating system.

The product is a Smart Card Integrated Circuit (IC) with the **MultiAppV1.0** platform as defined in Table 2.

TOE Components	Identification	Constructor
IC	P5CD144 version V0B	NXP
Platform	MultiAppV1.0	Gemalto

Table 2. Product component

The TOE defined in this Security Target is the MultiAppV1.0 platform. The TOE will be designed and produced in a secure environment and may be used by each citizen in a hostile environment.

The product is compliant with:

- Java Card 2.2.1
- Global Platform 2.1.1

1.3.1 TOE type

The product is a smartcard a module performing the interface between reader and the embedded chip. Other smart card product elements (such as holograms, security printing…) are outside the scope of this Security Target. The Target of Evaluation (TOE) is the Smart Card Integrated Circuit with Embedded Software in operation and in accordance to its functional specifications.

1.3.2 TOE boundaries

The TOE is composed of the IC and the software platform:

- The **P5CD144** IC which has been certified separately according to [ST/NXP] claiming [PP/BSI-0002]
- The **MultiAppV1.0** platform

The **TSFs** are composed of:

1. The platform MultiAppV1.0 that installs and supports the e-Identity applications.
2. The **P5CD144** IC

Figure 1 represents the TOE: the border of the TOE is represented by the un-continuous red lines.

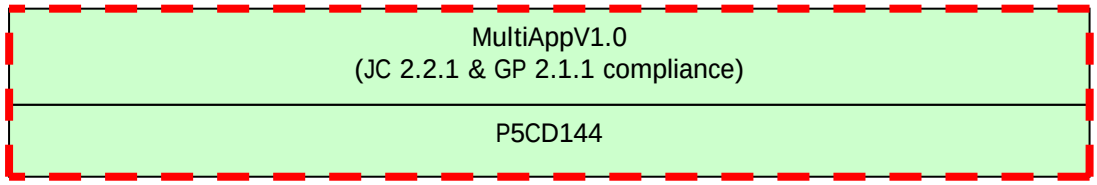

Figure 1. The TOE inside the product

1.4 References, Glossary and Abbreviations

Reference	Title
[CC-1]	Common Criteria for Information Technology Security Evaluation Evaluation Part 1: Introduction and general model, CCMB-2006-09-001, version 3.1, Rev. 1, September 2006
[CC-2]	Common Criteria for Information Technology Security Evaluation Evaluation Part 2: Security Functional Requirements, CCMB-2007-09-002, version 3.1, Rev. 2, September 2007
[CC-3]	Common Criteria for Information Technology security Evaluation Evaluation Part 3: Security Assurance Components, CCMB-2007-09-003, version 3.1, Rev. 2, September 2007

Reference	Title
[CEM]	Common Methodology for Information Technology Security Evaluation Evaluation Methodology, CCMB-2007-09-004, version 3.1, Rev. 2, September 2007.
[PP/BSI-0002]	Smart Card IC Platform Protection Profile, version 1.0, registered by BSI in 2001 under PP-BSI-0002, Eurosmart document (SSVG Protection Profile).
[PP/JCS]	Java Card System Protection Profile Version 1.0b, © Sun Microsystems Inc., August 2003.
[ST/NXP]	P5CD144/P5CN144/P5CC144 V0B Security Target Lite , Rev 1.0, @NXP, March 2007
[BSI/P5]	Certification Report for NXP Secure Smart Card Controller P5CD144V0B, P5CN144V0B and P5CC144V0B each with specific IC Dedicated Software. July 2007 © BSI. Ref: BSI-DSZ-CC-0411-2007
[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.
[ISO 7816-6]	Identification cards - Integrated circuit(s) cards with contacts, Part 6: Interindustry data elements.
[JCAPI221]	Java Card™ APIs specification version 2.2.1, Sun Microsystems, Inc, June 23, 2003.
[JCAPN221]	Application Programming Notes for the Java Card™ Platform, ©Sun Microsystems, Inc, version 2.2.1, October 2003.
[JCRE221]	Java Card™ Runtime Environment Specification version 2.2.1, ©Sun Microsystems, Inc, 2003.
[JCVM221]	Java Card™ Virtual Machine Specification version 2.2.1, ©Sun Microsystems, Inc, 2003.
[JC_BV22]	Java Card™ 2.2 Off-Card verifier (White Paper), ©Sun Microsystems, Inc, 2002.
[JVM]	The Java Virtual Machine Specification. Lindholm, Yellin. ISBN 0-201-43294-3.
[GP211]	Global Platform Card Specification – v2.1.1, March 2003.
[GUD_MultiApp]	MultiAppV1.0 Platform: Guidance Documents

2 TOE Description

MultiAppV1.0 platform is a smart card OS that complies with two major industry standards:

1. Sun's Java Card 2.2.1, which consists of the Java Card 2.2.1 Virtual Machine, Java Card 2.2.1 Runtime Environment and the Java Card 2.2.1 Application Programming Interface.
2. The GlobalPlatform Card Specification version 2.1.1.

MultiAppV1.0 contains the following components (see Figure 2):

The *Native Layer* that provides the basic card functionalities (memory management, I/O management and cryptographic libraries) with native interface with the dedicated IC. The cryptographic library includes TDES, RSA standard and CRT (up to 2048), ECDSA, ECDH, hashing (SHA-1, SHA-256), and RNG.
The *Java Card Runtime Environment*, which provides a secure framework for the execution of Java Card programs and data access management (firewall).
The *Java Card Virtual Machine*, which provides the secure interpretation of bytecodes.
The *API* including the standard Java Card API, the JCF API (Biometry) and Gemalto proprietary API.
The *Open Platform Card Manager*, which provides card, key and application management functions (contents and life-cycle) and security control.

The MultiAppV1.0 platform provides the following services:

1. Initialization of the Card Manager and management of the card life cycle,
2. Secure installation of the application under Card Manager control,
3. Extradition services[1] to allow several applications to share a dedicated security domain,
4. Deletion of applications under Card Manager control,
5. Secure operation of the applications through the API,
6. Card basic security services as follows:
   - Checking environmental operating conditions using information provided by the IC,
   - Checking life cycle consistency,
   - Ensuring the security of the PIN and cryptographic key objects,
   - Generating random number,
   - Handling secure data object and backup mechanisms,
   - Managing memory content,
   - Ensuring Java Card firewall mechanism.

Figure 2 shows the architecture of the MultiAppV1.0 platform inside the TOE, that is bounded by the red un-continuous lines.

[1] The GlobalPlatform Card Content extradition process is designed to allow the association, to a different Security Domain, of a previously installed Application (See Section 6.4.3 of [GP211]).

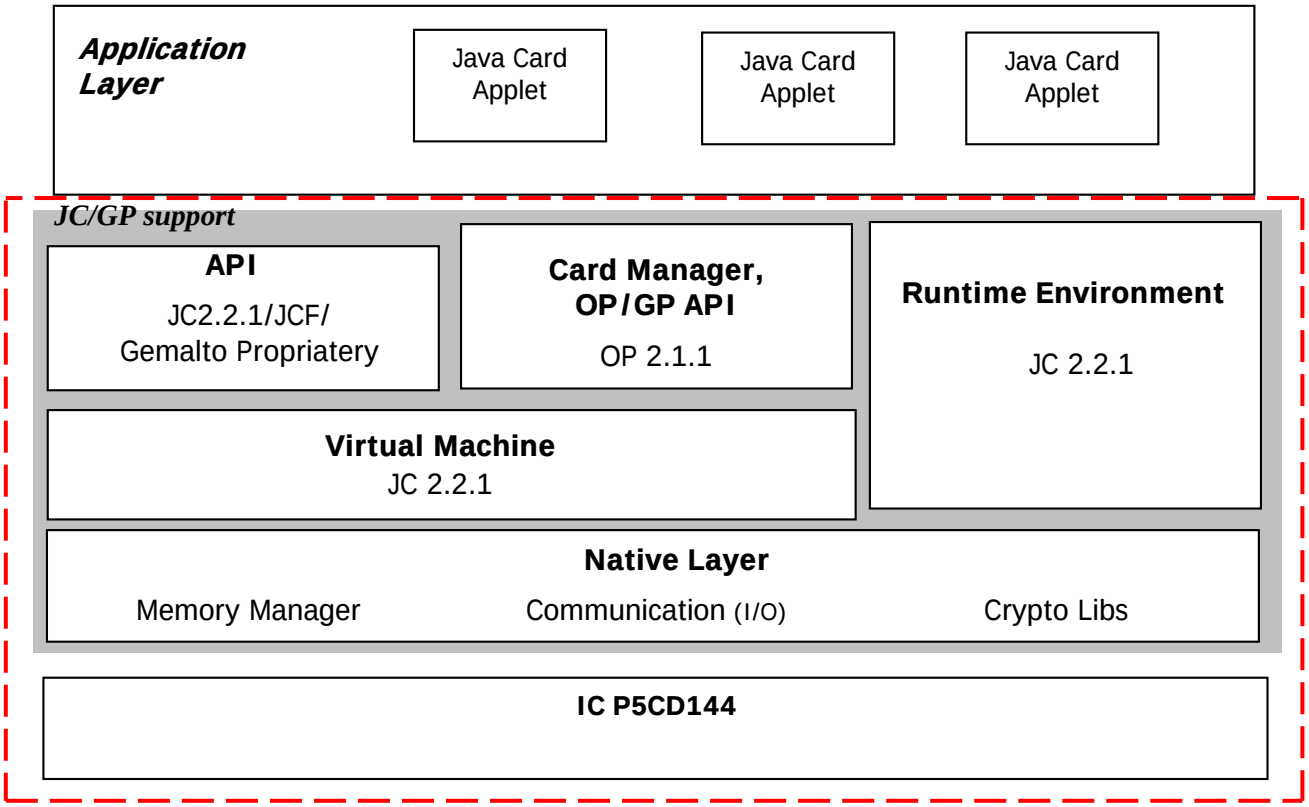

Figure 2. Architecture of MultiAppV1.0 Platform

2.1 TOE Life-cycle

The general life-cycle of a smart card product, as defined in [PP/BSI-0002], is split up into 7 phases as described in Table 3.

Phase 1	Smart Card software development	The SW developer is responsible for the development of the OS (the MultiAppV1.0 platform) and *the application*.
Phase 2	IC design, IC database construction and IC photomask fabrication	The IC manufacturer is responsible for these operations, taking as an input the embedded software data given by SW developer.
Phase 3	IC manufacturing and testing	The IC manufacturer is responsible for producing and testing the IC through three main steps: IC manufacturing, testing, and IC pre-personalization.
Phase 4	Module assembling and packaging	The smart card product manufacturer is responsible for assembling the module.
Phase 5	Smart card pre-personalization	The smart card product manufacturer is responsible for the smart card initialization, for installing the application (if needed) and for testing, and the smart card pre-personalization.
Phase 6	Smart card Personalization	The Personaliser is responsible for the personalization of *the application*. The application can also be installed in this phase.
Phase 7	Smart card Usage	The smart card issuer is responsible for the smart card product delivery to the smart card end-user, and for the end of life process.

Table 3. Smart Card Product Life cycle

For the current evaluation, the personalization and the final usage are not included in the TOE. Hence, the TOE life cycle only contains 5 phases as described in Figure 3. Some following remarks are added to explain this figure:

- The TOE is the product at the end of the phase 5.
- **Platform design and implementation** corresponds to the phase 1.
- **Hardware design** corresponds to the phase 2. In this phase, the MultiAppV1.0 platform is loaded (by masking) to the ROM.
- **Hardware manufacturing** corresponds to the phase 3.
- **Platform initialization** is done in the phase 5.

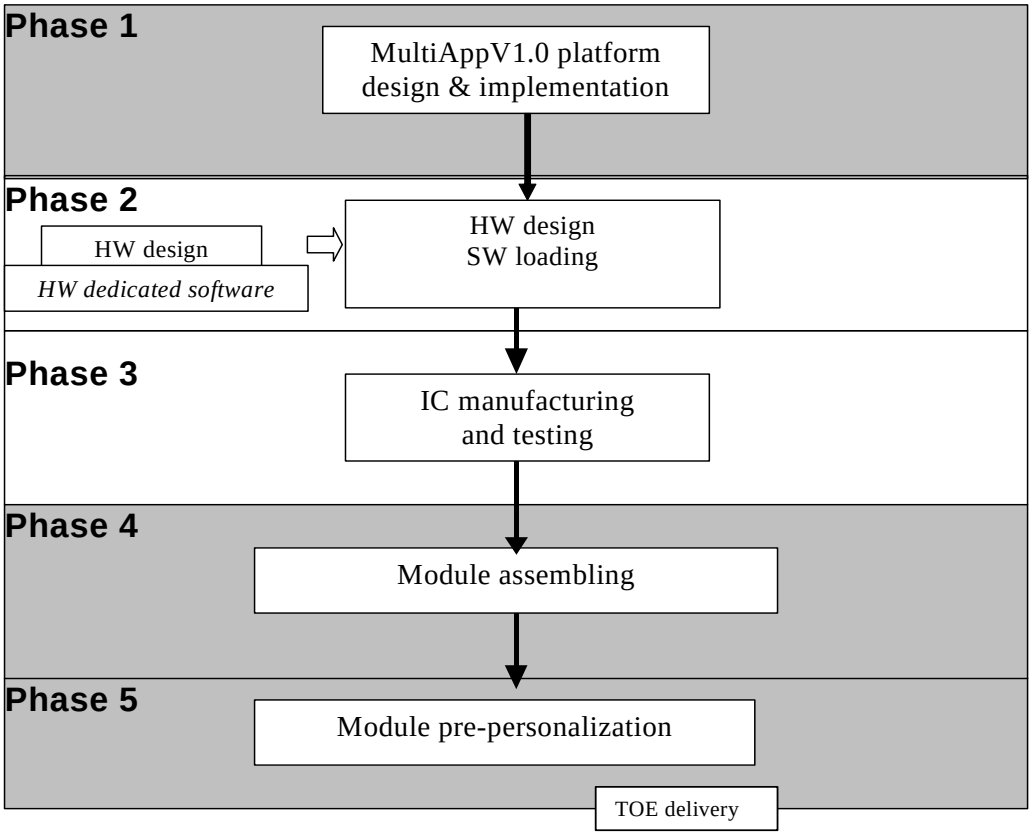

Figure 3. Product Life Cycle

The limits of the evaluation process correspond to the five phases (from 1 to 5). The limits includes the TOE under development, and the delivery between the phases.

These different phases may be performed at different sites. This implies that procedures on the delivery process of the TOE must exist and be applied for every delivery within a phase or between phases. This includes any kind of delivery performed from phase 1 to 5 to subsequent phases, including:

- Intermediate delivery of the TOE or the TOE under construction within a phase,
- Delivery of the TOE or the TOE under construction from one phase to the next.

These procedures must be compliant with the security assurance requirements developed in TOE Security Assurance Requirements (Section 6.2).

2.2 TOE Environment

The TOE described in this ST is developed in different places as indicated below:

IC design	NXP
SW Design (for MultiAppV1.0 only) including Pre-personalization design	Gemalto
IC manufacturing	NXP
Module assembling & packaging	Gemalto
Pre-personalization	Gemalto

The IC development and production are protected as decribed in the ST of the IC [ST/NXP]. A transport key protects the IC delivery from NXP to Gemalto. We are only interested below in the software aspect of the TOE.

In order to ensure security, the environment in which the development takes place must be made secure with access control tracing entries. Furthermore, it is important that all authorized personnel feel involved and fully understand the importance and the rigid implementation of the defined security procedures.

The development begins with the TOE specification. All parties in contact with sensitive information are required to abide by Non-disclosure Agreement (NDA).

Design and development of the embedded software then follows. The engineers use a secure computer system (preventing unauthorised access) to make the conception, design, implementation, and test performances. Storage of sensitive documents, databases on tapes, diskettes, and printed circuit layout information are in appropriately locked cupboards/safe. Of paramount importance also is the disposal of unwanted data (complete electronic erasures) and documents (e.g. shredding).

Testing, programming and deliveries of the TOE then take place. When these are done off-the-site, they must be transported and worked on in a secure environment with accountability and traceability of all (good and bad) products. During the electronic transfer of sensitive data, procedures must be established to ensure that the data arrive, only at the destination and is not accessible at intermediate stages (e.g. stored on a buffer server where system administrators make backup copies). It must also be ensured that transfer is done without modification or alteration.

During the TOE fabrication (phases 3 and 4), all people involved in storage and transportation operations must fully understand the importance of the defined security procedures. Moreover, the environment in which these operations take place must be secured.

The TOE initialization is performed by NXP (phase 3) and Gemalto (phase 4 and phase 5). In the production environment of the TOE, smart card pre-personalization takes place (phase 5). During smart card pre-personalization, the application data structure is created. This phase requires a secure environment, which guarantees the integrity and confidentiality of operations.

2.3 The actors and roles

The actors can be divided in:

Developers

The IC designer and Dedicated Software (DS) developer designs the chip and its DS. For this TOE, it is NXP.

The SW developer designs the OS (MultiAppV1.0) according to IC/DS specifications. For this TOE, it is GEMALTO.

Manufacturers

The IC manufacturer -or founder- designs the photomask, manufactures the IC with its DS and hardmask from the SW Developer. For this TOE, the founder is NXP.

The IC assembling manufacturer is responsible for assembling the modules using the ICs provided by the founder. For this TOE, the IC assembling manufacturer is GEMALTO.

The smart card product manufacturer (or card manufacturer) is responsible to pre-personalize the module. For this TOE, the smart card product manufacturer is GEMALTO.

2.4 TOE intended usage

The TOE is intended to be used as a platform to embedd the e-Identity applications written as Java Card applets. The usage of these applets and the conditions these applets shall satisfy are described in the TOE user's guide [GUD_MultiApp].

3 Conformance claims (ASE_CCL)

3.1 CC conformance claim

The compliance is assumed with Common Criteria version V3.1 (ISO 15408) ([CC-1], [CC-2], [CC-3]).

This product uses a certified IC conformant to the Protection Profile SSVG ([PP/BSI-0002]) from Eurosmart.

3.2 PP claim, Package claim

The assurance level claimed for this product is **EAL2**.

3.3 Conformance rationale

This ST is conformant with [CC-2].
This ST is conformant with [CC-3].
The [ST/NXP] refines the assets, threats, objectives and SFR of [PP/BSI-0002] (see BSI certification report [BSI/P5]).

3.4 PP reference

The Java Card protection profile is not claimed here. However, the present TOE is inspired by the TOE defined in [PP/JCS] for the **Minimal** configuration.

4 Security problem definition (ASE_SPD)

4.1 Assets

The platform assets are presented in the following table.

D. CODE	The code of the platform and the Java Card applets (applications).
D.OP_REGISTRY	Open Platform registry (see [GP211]) that contains the Card Manager (CM) data needed for card management operations.
D.ISD_KEYS	ISD keys, *i.e.* the Card Manager keys needed for the card personalization (phase 6 of the product lifecycle) including application loading, installing and initialization. Those include keys for authentication, encryption (ENC) and integrity (MAC).
D.PIN	User's PIN (to be protected from unauthorized disclosure and modification).
D.APP_KEYS	Cryptographic keys owned by the Java Card applets.
D.JAVA_OBJECT	Any Java data objects (owned by an application or by the platform)

4.2 Subjects

The platform subjects are presented in the following table:

S.Card_Manager	**The TOE entity acting on behalf of** the authorized platform administrator (e.g., the **Card Issuer**), that manages the card contents and controls application privileges.
S. Package	**Java Card packages** loaded on the platform and acts on behalf of the applet developer.
S.JCRE	**JCRE** acts with the "system priviledge" when accessing to D.JAVA_OBJECT

4.3 Threats

The threat agents are presented in the following table:

S.OFFCARD	**Attacker**. A human or process being located outside the TOE. The main goal of **S.OFFCARD** is to access application sensitive information. The attacker has a **basic potential attack** and **knows no secret.**

The platform threats are presented in the following table:

T.Plt_Integrity	Integrity of the platform data and code. **S.OFFCARD** tries to alter platform stored sensitive data (assets) or code to gain access to unauthorized data or operations. This threat concerns **D.ISD_KEYS**, **D.OP_REGISTRY, D.PIN, D.APP_KEYS,** and **D.CODE.**
T.Plt_Confidentiality	Confidentiality of platform data. **S.OFFCARD** tries to disclose platform-stored data to gain access to unauthorized operations.

	This threat concerns **D.ISD_KEYS, D.PIN, D.APP_KEYS.**
T.Plt_Install	**S.OFFCARD** fraudulently install an applet on the card. This concerns either the installation of an unauthorized applet or an attempt to induce a malfunction in the TOE through the installation process. This threat concerns applets installation and mainly **D.OP_REGISTRY**.
T.Plt_Execution	**S.OFFCARD** or **S.Package** executes code in order to gain illegal access to platform or applet resources. This threat deals with **D.CODE** and **D.JAVA_OBJECT** access.
T.Plt_Operate	**S.OFFCARD** or **S.Package** tries to modify the platform behavior by unauthorized or incorrect use of commands, or by producing malfunction conditions. This includes bad command, authentication bypass, in-secure state by insertion or interruption of session. This threat concerns all platform assets.

4.4 Organizational Security Policies (OSP)

The OSPs of the IC are described in its security target [ST/NXP]. The following table contains the OSP of the platform:

P.Plt_Support	The platform is conformant to Java Card 2.2.1 and GP 2.1.1 stadards and allows the e-Identity application (a Java Card applet) to operate in a secure environment. The platform shall ensure that only **S.Card_Manager** can perform the following secure operations: load, install and delete application. The platform shall provide the applications with a secure execution environment as well as a mechanism for secure data sharing. The platform shall provide the applications with cryptographic services, in particular, RSA (up to 2048), ECDSA, ECDH, AES, TDES, SHA-1, SHA-256, RNG.

4.5 Assumption

The assumption is presented in the following table:

A.Applets	All applets to be added on the platform shall be successfully verified by an off-card Java Card Bytecode Verifier [JC_BV22].
A.Native	All applets to be added on the platform shall not contain native code.

5 Security objectives (ASE_OBJ)

5.1 Security objectives for the TOE

The security objectives of the IC are described in its security target [ST/NXP]. The security objectives of the MultiAppV1.0 platform are presented in the following table:

OT.SID	The TOE shall uniquely identify every subject (e.g., Java Card package, Card Manager) before granting access to any crucial service.
OT.Plt_Integrity	The TOE shall ensure that the sensitive data (i.e., assets) stored in the memory are protected against corruption or unauthorized modification. The TOE shall provide means to verify the integrity of its code.
OT.Plt_Confidentiality	The TOE shall provide mechanisms to securely manage *D.PIN* to avoid unauthorized access, disclosure or snooping. The TOE shall provide mechanisms to securely manage *D.ISD_KEYS and D.APP_KEYS* to avoid unauthorized access, disclosure or snooping.
OT.Plt_Reallocation	The TOE shall ensure that the re-allocation of a memory block does not disclose the sensitive information previously stored in that block.
OT.Plt_Install	The TOE shall ensure that only **S.CARD_MANAGER** is allowed to load, install and delete applets. The TOE shall provide a means to securely perform the applet initialization.
OT.Plt_Execution	The TOE shall ensure that only **S.Card_Manager** is allowed to manage the card content through the dedicated commands.
OT.Plt_Firewall	The TOE shall ensure controlled sharing of data containers owned by applets of different packages, and between applets and the TSFs.
OT.Shrd_Var_Integ	The TOE shall ensure that only the currently selected application may grant write access to a data memory area that is shared by all applications, like the APDU buffer, the byte array used for the invocation of the process method of the selected applet, or any public global variable exported by the API. Even though the memory area is shared by all applications, the TOE shall restrict the possibility of getting a reference to such memory area to the application that has been selected for execution. The selected application may decide to temporarily hand over the reference to other applications at its own risk, but the TOE shall prevent those applications from storing the reference as part of their persistent states.
OT.Plt_Operate	The TOE shall ensure correct operation of its security function and guarantee that the environment, in which the application operates, is safe. The TOE shall provide appropriate feedback information upon detection of potential violation.
OT.Plt_Support	The TOE must provide a means to execute a set of operations atomically. The TOE shall provide cryptographic services such as RSA (up to 2048), ECDSA, ECDH, AES, TDES, SHA-1, SHA-256, RNG.

5.2 Security objectives for the operational environment

OE.Applet	The applets added on the TOE shall be successfully verified by an off-card Java Card Bytecode Verifier [JC_BV22].
OE.Native	The applets added on the TOE shall only use native code that is part of the platform API (Java Card API, or additional proprietary API).

5.3 Security objectives rationale

Table 4 shows the rationale of the security objectives.

Threats, Assumptions, OPSs *vs* **Security Objectives**	**OT.SID**	**OT.Plt_Integrity**	**OT.Plt_Confidentiality**	**OT.Plt_Reallocation**	**OT.Plt_Install**	**OT.Plt_Execution**	**OT.Plt_Firewall**	**OT.Shrd_Var_Integ**	**OT.Plt_Operate**	**OT.Plt_Support**	**OE.Applet**	**OE.Native**
T.Plt_Integrity		X										
T.Plt_Confidentiality			X	X								
T.Plt_Install	X				X							
T.Plt_Execution	X					X	X	X				
T.Plt_Operate	X								X			
P.Plt_Support	X	X	X	X	X	X	X	X	X	X		
A.Applet											X	
A.Native												X

Table 4. Security objectives rationale

All threats, OPS and assumptions are covered by the security objectives as follows:

- **T.Plt_Integrity** is countered by **OT. Plt_Integrity** that ensures the protection of data stored in the memory, against corruption or unauthorized modification and the code integrity check.
- **T.Plt_Confidentiality** is countered by **OT.Plt_Confidentiality** and **OT.Plt_Reallocation**
  - **OT.Plt_Confidentiality** ensures the confidentiality of the data when being stored or used.
  - **OT.Plt_Reallocation** ensures that the sensitive data is not disclosed during re-allocation.
- **T.Plt_Install** is countered by **OT. SID** and **OT.Plt_Install**
  - **OT. SID** ensures the identification of the Card Manager before the installation process.
  - **OT.Plt_Install** ensures that only the (authenticated) Card Manager is allowed to install new applications.
- **T.Plt_Execution** is countered by **OT.SID, OT.Plt_Execution, OT.Plt_Firewall** and **OT.Shrd_Var_Integ**
  - **OT. SID** ensures any subject is identified before excuting a code or access to a Java object.
  - **OT.Plt_Execution** prevents the access from un-authorized subjects to platform and applet resources.

- **OT.Plt_Firewall** and **OT.Shrd_Var_Integ** ensure that only authorized subjects can access to the Java objects.

– **T.Plt_Operate** is countered by **O.SID** and **O.Plt_Operate**
- **OT. SID** ensures any subject is identified before executing the crucial commands that may modify the platform behavior.
- **O.Plt_Operate** ensures that correct operation of security functions that prevent the execution of un-authorized commands, the authentication bypass, and keep the TOE in a secure state.

– **P.Plt_Support** is enforced by all security objectives, in particular:
- **OT.SID, OT.Plt_Execution** and **OT_Plt_Install** ensure that only **S.Card_Manager** can perform the secure operations such as loading, installing and deleting application.
- **OT.Plt_Firewall** and **OT.Shrd_Var_Integ** provide a mechanism for secure data sharing.
- **OT.Plt_Support** provides the applications with cryptographic services. **OT.Plt_Support** also enforces the secure execution environment by providing a means to execute atomically a set of operations.
- **OT. Plt_Integrity, OT.Plt_Confidentiality, and OT.Plt_Reallocation** protect the integrity and the confidentiality of the sensitive data of the applications and hence, enforce the secure execution environment for the applications.
- **OT.Plt_Operate** ensures the correct operation of the security functions, and hence provides a secure execution environment that is conformant to JC 2.2.1 and GP 2.1.1 standards.

– **A.Applet** is covered by **OE.Applet** that addresses the conformance of embedded applet (with respect to Java Card specification), which is outside the scope of the TOE.

– **A.Native** is covered by **OE.Native** that addresses the use of native code in the applications and covers. Actually, the Java Card platform cannot ensure the security of external native code.

6 Security requirements (ASE_REQ)

6.1 TOE security functional requirements

[ST/NXP] deals with the security functional requirements of [PP/BSI-0002]. In this section, we only provide the security functional requirements of the MultiAppV1.0 platform.

6.1.1 Platform security functional requirements list

Table 5 provides the list of SFRs fulfilled by the TOE. In the rest of Section 6.1, the detail of each SFR is described. For improving the readability, in the definition of SFRs, the normal typesetting comes from the CC standard [CC-2] while the **bold text** represents the specific elements (both selection and assignment) of this evaluation (e.g., the implemented cryptographic mechanisms). An *application note* is added if necessary to explain the specific features of the SFR. A ***REFINEMENT*** is added if the SFR is needed to be adapted to the TOE.

Identification	DESCRIPTION
FAU	**Security audit**
FAU_ARP.1	Security alarms
FCS	**Cryptographic support**
FCS_COP.1	Cryptographic operation
FCS_CKM.1	Cryptographic key generation
FCS_CKM.4	Cryptographic key destruction
FDP	**User data protection**
FDP_ACC.1	Subset Access control
FDP_ACC.2	Complete Access control
FDP_ACF.1	Security attributes based access control
FDP_IFC.1	Subset information flow control
FDP_IFF.1	Simple security attributes
FDP_RIP.1	Subset residual information protection
FDP_SDI.2	Stored data integrity monitoring and action
FDP_UIT.1	Basic data exchange intergrity
FIA	**Identification and Authentication**
FIA_ATD.1	User attribute definition
FIA_UAU.1	Timing of authentication
FIA_UID.1	Timing of identification
FIA_USB.1	User-subject binding
FMT	**Security management**
FMT_MOF.1	Management of security function behavior
FMT_MSA.1	Management of security attributes
FMT_MSA.3	Static attribute initialization
FMT_MTD.1	Management of TSF data
FMT_SMF.1	Specification of Management Function
FMT_SMR.1	Security roles
FPT	**Protection of the TOE Security function**
FPT_FLS.1	Failure with preservation of secure state
FPT_PHP.1	Passive detection of physical attack
FPT_PHP.3	Resistance to physical attack
FPT_TDC.1	Inter-TSF Data consistency
FTP	**Trusted path/Channel**
FTP_ITC.1	Inter-TSF Trusted Channel

FTP_TRP.1	Trusted Path

Table 5. Platform security functional requirements list

6.1.2 Security audits (FAU)

6.1.2.1 FAU_ARP.1 Security alarms

FAU_ARP.1.1	The TSF shall take one of the following **disruptive actions** upon detection of a potential security violation. **List of disruptive actions:** 1. Reset the card and clear all volatile memory. 2. Block the action that produced the security violation and throw an exception. 3. Terminate the card (put the card life cycle to TERMINATED) 4. Mute the card. ***REFINEMENT*** **Potential security violation is refined to one of the following events:** a. Security error reported by a IC sensor: action (1) b. Life cycle state inconsistency (D.OP_REGISTRY): action (3) and (4) if the fault counter attains its maximal value c. Integrity errors on D.ISD_KEYS, D.PIN, D.APP_KEYS: action (3) and (4) if the fault counter attains its maximal value d. Illegal Access to D.JAVA_OBJECT: action (2) using a **SecurityException** e. Unavailability of resources audited through the object allocation mechanism: action (2) using a **SystemException**

6.1.3 Cryptographic support (FCS)

6.1.3.1 FCS_COP.1 Cryptographic operations

FCS_COP.1.1 / TDES	The TSF shall perform **TDES encryption and decryption** in accordance with a specified cryptographic algorithm **TDES-CBC, TDES-EBC** and cryptographic key sizes **112 bits for TDES 2 keys, 168 bits for TDES 3 keys** that meet the following: **[FIPS 46-3]**. *Application note:* The TOE can also encrypt and decrypt using DES algorithm with 56 bits key, but this is to be considered as a service. The DES algorithm is no longer considered as resistant to high level attacks.
FCS_COP.1.1 / AES	The TSF shall perform **AES encryption and decryption** in accordance with a specified cryptographic algorithm **AES** and cryptographic key sizes **128 bits** that meet the following: **[FIPS-197].**
FCS_COP.1.1 / RSA	The TSF shall perform **RSA encryption and descryption** in accordance with a specified cryptographic algorithm **RSA** and cryptographic key sizes **1024, 1152, 1280, 1536 and 2048 bits** that meet the following: **standard and CRT**.
FCS_COP.1.1 / RSA-OAEP	The TSF shall perform RSA encryption and descryption in accordance with a specified cryptographic algorithm **RSA-OAEP** and cryptographic key sizes **1024, 1152, 1280, 1536 and 2048 bits** that meet the following: **no standard**.
FCS_COP.1.1 / CORRESP	The TSF shall perform **public key / private key correspondence verification** in accordance with a specified cryptographic algorithm **RSA key computation** and cryptographic key sizes **1024, 1152, 1280, 1536 or 2048 bits** that meet the following: **no standard**.

FCS_COP.1.1 / SHA	The TSF shall perform **secure hashing** in accordance with the specified cryptographic algorithms **SHA-1, SHA-256** and cryptographic key sizes **none** that meet the following: **[FIPS 180-2]**.
FCS_COP.1.1 / RNG	The TSF shall perform **Random Number Generation** in accordance with a specified cryptographic algorithm **Random Number Generator** and cryptographic key sizes **None** that meet the following: **ANSI X9.17 (Appendix C).**
FCS_COP.1.1 / MAC	The TSF shall perform **secure messaging – message authentication code** in accordance with a specified cryptographic algorithm **TDES** and cryptographic key sizes **128 bits** that meet the following specification: **Annexes E.4.4 and E.4.5 of [GP211].**
FCS_COP.1.1 / ECDSA	The TSF shall perform **electronic signature-generation** in accordance with a specified cryptographic algorithm **ECDSA** and cryptographic key sizes **224, 256, and 384 bits** that meet the following: **[ANSI X9.62]**.
FCS_COP.1.1 / ECDH	The TSF shall perform **key agreement** in accordance with a specified cryptographic algorithm **ECDH** and cryptographic key sizes **224, 256, and 384 bits** that meet the following: **[ANSI X9.63]**.

6.1.3.2 FCS_CKM.1 Cryptographic key generation

FCS_CKM.1.1 / TDES	The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **TDES for the generation of session keys** and specified cryptographic key sizes **128 bits** that meet the following standards: **Annexe E.4.1 of [GP211].**
FCS_CKM.1.1 / RSA	The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **RSA key generation** and specified cryptographic key sizes **1024, 1152, 1280, 1536 and 2048 bits** that meet the following: **RSA standard.**
FCS_CKM.1.1 / ECC	The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **ECC key generation** and specified cryptographic key sizes **224, 256, and 384 bits** that meet the following: **[ANSI X9.62].**

6.1.3.3 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: **overwrite the keys** that meets the following: **no standard**.

6.1.4 User data protection (FDP)

6.1.4.1 FDP_ACC.1 Subset access control

FDP_ACC.1.1 / Card Manager SFP	The TSF shall enforce the **Card Manager SFP** on the following list of subjects, objects and operations.	
Subjects	**Objects**	**Operations**
S.Card_Manager	D.OP_REGISTRY	Applet installation and deletion Change the Card Life Cycle state Change the Application Life Cycle state

6.1.4.2 FDP_ACC.2 Complete access control

FDP_ACC.2.1 / Firewall SFP	The TSF shall enforce the **Firewall SFP** on **S.Package, S.JCRE, D.JAVA_OBJECT** and all operations among subjects and objects covered by

	the SFP.
Operation	**Description**
OP.ARRAY_ACCESS (D.JAVA_OBJECT, field)	Read/Write an array component.
OP.INSTANCE_FIELD (D.JAVA_OBJECT, field)	Read/Write a field of an instance of a class in the Java programming language
OP.INVK_VIRTUAL (D.JAVA_OBJECT, method, arg1,...)	Invoke a virtual method (either on a class instance or an array object)
OP.INVK_INTERFACE (D.JAVA_OBJECT, method, arg1,...)	Invoke an interface method.
OP.THROW (D.JAVA_OBJECT)	Throwing of an object (**athrow**).
OP.TYPE_ACCESS (D.JAVA_OBJECT, class)	Invoke **checkcast** or **instanceof** on an object.
OP.JAVA (...)	Any access in the sense of [JCRE221], §6.2.8.
OP.CREATE (Sharing, LifeTime)	Creation of an object (**new** or **makeTransient** call).

Operations (prefixed with " *OP* ") of this policy are described above. Each operation has a specific number of parameters given between brackets, among which there is the "**accessed object** ", the first one, when applicable. Parameters may be seen as security attributes that are under the control of the subject performing the operation. Note that accessing array's components of a **static** array, and more generally fields and methods of **static** objects, is an access to the corresponding **D.JAVA_OBJECT**.

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.

6.1.4.3 FDP_ACF.1 Security attributes based access control

FDP_ACF.1.1/ Card_Manager SFP	The TSF shall enforce the **Card Manager SFP** to objects based on **following attributes.**	
Subject/object	**Attribute**	**Values**
S.Card_Manager	Authentication	Yes, No
	Secure Channel	Open, Not Open
D.OP_REGISTRY	Card Life Cycle state	OP_READY, INITIALIZED, SECURED, CARD_BLOCKED, TERMINATED

FDP_ACF.1.2/ Card_Manager SFP	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed.
Attributes	**Rules**
Authentication Secure Channel Card Life Cycle state Available Load	The **Secure Channel** is set to Open only if Card Manager has been correctly authenticated and **Authentication [Card Manager]** is set to Yes. Operations on applications are allowed only if **Card life Cycle state** is set to OP_READY, INITIALIZED or SECURED and if Card Manager has been correctly authenticated with **Authentication [Card Manager]** set to Yes. Only Card Manager correctly authenticated with **Authentication [Card Manager]** set to Yes is allowed to update **D.OP_REGISTRY** during Applet Install/Delete. Only Card Manager correctly authenticated with **Authentication [Card Manager]** set to Yes is allowed to set the Applet Life Cycle in **OP_REGISTRY** to INSTALL.

FDP_ACF.1.3 / Card Manager SFP	The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: **None.**
FDP_ACF.1.4 / Card Manager SFP	The TSF shall explicitly deny access of subjects to objects based on the rules**: None.**

FDP_ACF.1.1 / Firewall SFP	The TSF shall enforce the **Firewall SFP** to objects based on the following: **(1) the security attributes of the covered subjects and objects, (2) the currently active context and (3) the SELECTed applet c**ontext.	
Subject / object	**Attribute**	**Values**
S.JCRE	None	
S.Package	Context	Package AID or “JCRE”
D.JAVA_OBJECT	Context	Package AID or “JCRE”
	Sharing	Standard (both filed sand methods are under firewall policy), or Shareable Interface Object (SIO), or JCRE Entry Point (temporary or permanent), or Global Array
	Lifetime	CLEAR_ON_DESELECT or PERSISTENT
	SELECTed applet context	Package AID or “None”

Both "the currently active context" and "the SELECTed applet context" are internal security attributes to the platform, that is, not attached to any specific object or subject. The currently active context is defined in Section 6.1.2.1 of [JCRE221]. The SELECted applet context is the context of the selected applet and so, must be either a package AID or “None” (when no applet is selected).

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

- **R.JAVA.1** ([JCRE221] §6.2.8) An S.PACKAGE may freely perform any of OP.ARRAY_ACCESS, OP.INSTANCE_FIELD, OP.INVK_VIRTUAL, OP.INVK_INTERFACE, OP.THROW or OP.TYPE_ACCESS upon any *D.JAVA_OBJECT* whose Sharing attribute has value "JCRE Entry Point" or "Global Array".
- **R.JAVA.2** ([JCRE221] §6.2.8) An S.PACKAGE may freely perform any of OP.ARRAY_ACCESS, OP.INSTANCE_FIELD, OP.INVK_VIRTUAL, OP.INVK_INTERFACE or OP.THROW upon any D.JAVA_OBJECT whose Sharing attribute has value "Standard" and whose Lifetime attribute has value "PERSISTENT" only if D.JAVA_OBJECT 's Context attribute has the same value as the active context.
- **R.JAVA.3** ([JCRE221] §6.2.8.10) An S.PACKAGE may perform OP.TYPE_ACCESS upon an D.JAVA_OBJECT whose Sharing attribute has value "SIO" only if D.JAVA_OBJECT 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** ([JCRE221]§6.2.8.6) An S.PACKAGE may perform OP.INVK_INTERFACE upon an D.JAVA_OBJECT whose Sharing attribute has the value "SIO" only if the invoked interface method extends the Shareable interface.

FDP_ACF.1.3 / Firewall SFP	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: **The subject S.JCRE can freely perform OP.JAVA*(...)* and OP.CREATE, with the exception given in FDP_ACF.1.4 / Firewall SFP**.

FDP_ACF.1.4 / Firewall SFP	The TSF shall explicitly deny access of subjects to objects based on the **rules:** **1. Any subject with OP.JAVA upon an D.JAVA_OBJECT whose LifeTime attribute has value "CLEAR_ON_DESELECT" if D.JAVA_OBJECT 's Context attribute is not the same as the SELECTed applet Context.** 2. **Any subject with OP.CREATE and a "CLEAR_ON_DESELECT" LifeTime parameter if the active context is not the same as the SELECTed applet Context**.

6.1.4.4 FDP_IFC.1 Subset information flow control

FDP_IFC.1.1 / JCVM SFP	The TSF shall enforce the **JCVM information flow control SFP** on **the following subjects, information and operations**.
Subject / Information	**Description**
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.
I.DATA	JCVM Reference Data: objectref address of temporary JCRE Entry Point objects and global arrays.
Operation	Description
OP.PUT(S_1, S_2, I)	Transfer a piece of information / from *S_1* to *S_2*.

Application note: References of temporary *JCRE entry points*, which cannot be stored in *class* variables, instance variables or array components, are transferred from the internal memory of the *JCRE* (TSF data) to some stack through specific APIs (*JCRE* owned exceptions) or *JCRE* invoked methods (such as the **process(APDU apdu)**); these are causes of *OP.PUT(S_1,S_2,I)* operations as well.

6.1.4.5 FDP_IFF.1 Simple security attributes

FDP_IFF.1.1 / JCVM SFP	The TSF shall enforce the **JCVM information flow control SFP** based on the following types of subject and information security attributes: **S.LOCAL, S.MEMBER, I.DATA and the currently active context**.

FDP_IFF.1.2 / JCVM SFP	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(S_1, S.MEMBER, I) is allowed if and only if the active context is "JCRE"; other OP.PUT operations are allowed regardless of the active context's value**.

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

FDP_IFF.1.4 / JCVM SFP	The TSF shall explicitly authorise an information flow based on the following rules: **all JCRE Permanent Entry Point Object may be stored in a S.MEMBER**.

FDP_IFF.1.5 / JCVM SFP	The TSF shall explicitly deny an information flow based on the following rules: **the storage of the reference of an object with attribute JCRE Temporary Entry Point Object or Global Array in a static field, instance field or array element is forbidden**.

6.1.4.6 FDP_RIP.1 Subset residual information protection

FDP_RIP.1.1	The TSF shall ensure that any previous information content of a resource is made unavailable upon the **deallocation of resource** for the following objects: • **D.ISD_KEYS** • **D.APP_KEYS** • **D.PIN**

6.1.4.7 FDP_SDI.2 Stored data integrity monitoring and action

The following data persistently stored by the TOE have the user attribute "**integrity checked persistent stored data**":

- KEYS: D.ISD_KEYS, D.APP_KEYS, D.PIN
- Card Life Cycle state (in D.OP_REGISTRY)
- Applet Life Cycle state (in D.OP_REGISTRY)

FDP_SDI.2.1/ KEYS	The TSF shall monitor user data stored in containers controlled by the TSF for **any inconsistency between D.ISD_KEYS, D.APP_KEYS, D.PIN and its checksum** on all objects, based on the following attributes: **integrity checked persistent stored data.**
FDP_SDI.2.2/ KEYS	Upon detection of a data integrity error, the TSF shall: • **Prohibit the use of the altered data** • **Mute the card** • **The fault counter is incremented by 1. If this counter attains its maximal value (i.e., 3), then the card is terminated.**

FDP_SDI.2.1/ Card_life_cycle	The TSF shall monitor user data stored in containers controlled by the TSF for **any inconsistency between the Card Life Cycle state (part of D.OP_REGISTRY) and its checksum** on all objects, based on the following attributes: **integrity checked persistent stored data.**
FDP_SDI.2.2/ Card_life_cycle	Upon detection of a data integrity error, the TSF shall: • **Prohibit the use of the altered data** • **Mute the card** • **The fault counter is incremented by 1. If this counter attains its maximal value (i.e., 3), then the card is terminated.**

FDP_SDI.2.1/ Applet_life_cycle	The TSF shall monitor user data stored in containers controlled by the TSF for **any inconsistency between the Applet Life Cycle state (part of D.OP_REGISTRY) and its checksum** on all objects, based on the following attributes: **integrity checked persistent stored data.**
FDP_SDI.2.2/ Applet_life_cycle	Upon detection of a data integrity error, the TSF shall: • **Prohibit the use of the altered data** • **Mute the card** • **The fault counter is incremented by 1. If this counter attaineds its maximal value (i.e., 3), then the card is terminated.**

6.1.4.8 FDP_UIT.1 Data exchange integrity

FDP_UIT.1.1	The TSF shall enforce the **Card Manager SFP,** to be able to **transmit** and **receive** objects in a manner protected from

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

6.1.5 Identification and Authentication (FIA)

6.1.5.1 FIA_ATD.1 User attribute definition

FIA_ATD.1.1	The TSF shall maintain the following list of security attributes belonging to individual users: - **Authentication** - **Context (i.e. Package AID)**

6.1.5.2 FIA_UAU.1 Timing of authentication

FIA_UAU.1.1	The TSF shall allow **the following TSF mediated actions (APDU commands from [GP211])** on behalf of the user to be performed before the user is authenticated: – **Get Data** – **Select** – **Initialize Update** – **Manage Channel**
FIA_UAU.1.2	The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

Application note: the authentication is only relative to the Card Manager. TSF mediated actions on behalf of the applications are not controlled by this authentication.

6.1.5.3 FIA_UID.1 Timing of identification

FIA_UID.1.1	The TSF shall allow **the following TSF mediated actions (APDU commands from [GP211])** on behalf of the user to be performed before the user is identified: – **Manage Channel**
FIA_UID.1.2	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 TSF-mediated actions are the APDU commands from [GP211]. 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.

6.1.5.4 FIA_USB.1 User-subject binding

FIA_USB.1.1	The TSF shall associate the user security attributes with subjects acting on behalf of that user.
FIA_USB.1.2	The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users: **None**.
FIA_USB.1.3	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 (dependencies): the security attributes are listed in FIA_ATD.1.

6.1.6 Security management (FMT)

6.1.6.1 FMT_MOF.1 Management of security function behavior

FMT_MOF.1.1	The TSF shall restrict the ability to **modify the behavior of** the functions **listed below** to **S.Card_Manager:** – **Load/Install/Delete application** – **Update D.ISD.KEY**

Application Note: S.Card_Manager may assign a delegated security domain (at the installation of this SD) that represents S.Card_Manager in loading and installating an application.

6.1.6.2 FMT_MSA.1 Management of security attributes

FMT_MSA.1.1 / Card Manager	The TSF shall enforce the **Card Manager SFP** to restrict the ability to **perform the following operations on** the security attributes **defined below** to the **Card manager.**

Object	Security attribute	Operation	SFP	Role
			See FDP_ACF.1	See FMT_SMR.1
D.OP_REGISTRY	Card Life Cycle state	Modify	Card Manager	Card Manager (phase 6)

Application Note: The delegated security domain (by S.Card_Manager) has the same role as S.Card_Manager.

FMT_MSA.1.1 / JCRE	The TSF shall enforce the **FIREWALL SFP and the JCVM SFP** to restrict the ability to **modify** the security attributes **the active context and the SELECTed applet Context** to **the JCRE (S.JCRE)**.

6.1.6.3 FMT_MSA.3 Static attribute initialization

FMT_MSA.3.1	The TSF shall enforce the **Card Manager SFP,** the **Firewall SFP,** and the **JCVM SFP** to provide **restrictive** default values for security attributes that are used to enforce the SFP.
FMT_MSA.3.2	The TSF shall allow **none** to specify alternative initial values to override the default values when an object or information is created.

6.1.6.4 FMT_MTD.1 Management of TSF data

FMT_MTD.1.1	The TSF shall restrict the ability to **access or modify** the **following TSF data** to **the Card Manager role: D.ISD_KEY.**

6.1.6.5 FMT_SMF.1 Specification of Management function

FMT_SMF.1.1	The TSF shall be capable of performing the following security management functions: – **Delete application** – **Load application** – **Install application** – **Update D.ISD.KEY** – **Modify D.OP_REGISTRY**

6.1.6.6 FMT_SMR.1 Security roles

FMT_SMR.1.1	The TSF shall maintain the roles **defined in the following list.** **<u>The roles list</u>:** **1. The Card Manager (Administrator) role** The Card manager is the in card representative of the Card Personalizer / Card Issuer The Card Manager may be in charge of application install operation and for setting the application state to INSTALLED and SELECTABLE, then for setting the Card Life Cycle state to SECURED. The Card Manager may be in charge of deleting an application. **2. The Application User role** After application installation, the platform sees the application as users. The access to platform resources is granted according to Java Card firewall access conditions.
FMT_SMR.1.2	The TSF shall be able to associate users with roles.

6.1.7 Protection of the TSF (FPT)

6.1.7.1 FPT_FLS.1 Failure with preservation of secure state

FPT_FLS.1.1	The TSF shall preserve a secure state when the following types of failures occur: **power shortage, over and under voltage, over and under clock frequency, over and under temperature, integrity problems**, **unexpected abortion of the execution of the TSF due to external events.**

6.1.7.2 FPT_PHP.1 Passive detection of physical attack

FPT_PHP.1.1	The TSF shall provide unambiguous detection of physical tampering that might compromise the TSF.
FPT_PHP.1.2	The TSF shall provide the capability to determine whether physical tampering with the TSF's devices or TSF's elements has occurred.

6.1.7.3 FPT_PHP.3 Resistance to physical attack

FPT_PHP.3.1	The TSF shall resist **voltage, clock frequency and temperature out of bounds as well as penetration attacks** to the **integrated circuit** by responding automatically such that the SFRs are always enforced.

6.1.7.4 FPT_TDC.1 Inter-TSF data consistency

FPT_TDC.1.1	The TSF shall provide the capability to consistently interpret **the CAP files (shared between the Byte Code Verifier and the TOE), the bytecode and its data arguments (shared between applets and API packages)** when shared between the TSF and another trusted IT product.
FPT_TDC.1.2	The TSF shall use **the following rules** when interpreting the TSF data from another trusted IT product. – **The [JCVM221] specification** – **Reference export files** – **The [ISO 7816-6] rules** – **The [GP211] specification**

6.1.8 Trusted path/channels (FTP)

6.1.8.1 FTP_ITC.1 Inter-TSF trusted channel

FTP_ITC.1.1 / D.ISD_KEYS load	The TSF shall provide a communication channel between itself and a remote 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 / D.ISD_KEYS load	The TSF shall permit **the remote trusted IT product** to initiate communication via the trusted channel.
FTP_ITC.1.3 / D.ISD_KEYS load	The TSF or the trusted IT shall initiate communication via the trusted channel for **D.ISD_KEY load**.

6.1.8.2 FTP_TRP.1 Trusted path

FTP_TRP.1.1	The TSF shall provide a communication path between itself and **local** users that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from modification or disclosure.
FTP_TRP.1.2	The TSF shall permit **local users** to initiate communication via the trusted path.
FTP_TRP.1.3	The TSF shall require the use of the trusted path for **application Install operation**.

6.2 TOE security assurance requirements

The TOE security assurance requirements define the assurance requirements for the TOE using only assurance components drawn from [CC-3]. The assurance level required for the TOE is **EAL2.**

6.2.1 TOE security assurance requirements list

Table 6 contains the list of security assurance requirements needed for the TOE.

Identification	Description
ASE	**Security Target Evaluation**
ASE_CCL.1	Conformance claims
ASE_ECD.1	Extended components definition
ASE_INT.1	ST introduction
ASE_OBJ.2	Security objectives
ASE_REQ.2	Derived security requirements
ASE_SPD.1	Security problem definition
ASE_TSS.1	TOE summary specification
ADV	**Development**
ADV_ARC.1	Security architecture description
ADV_FSP.2	Security-enforcing functional specification
ADV_TDS.1	Basic design
AGD	**Guidance documents**
AGD_OPE.1	Operational user guidance
AGD_PRE.1	Preparative procedures
ALC	**Life cycle support**
ALC_CMC.2	Use of a configuration management system
ALC_ CMS.2	Parts of the TOE configuration management coverage
ALC_DEL.1	Delivery procedures
ATE	**Tests**
ATE_COV.1	Evidence of coverage
ATE_FUN.1	Functional testing
ATE_IND.2	Independent testing – sample, including an audit of the developer's tests (sampling)
AVA	**Vulnerability assessment**
AVA_VAN.2	Vunerability analysis

Table 6. TOE security assurance requirements list

7 TOE summary specification (ASE_TSS)

This section describes the seucurity functions of the platform. The security functions of the IC are described in its security target [ST/NXP].

7.1 TOE security functions

SF_CARD_AUTHENTICATION	Card authentication
SF_CARD_CRYPTO	Card cryptographic algorithm & key management
SF_CARD_EMANATION	Emanation protection
SF_CARD_INTEGRITY	Card objects integrity
SF_CARD_MGR	Card Manager
SF_CARD_PROTECT	Card operation protection
SF_CARD_SECURE_MESSAGING	Card Secure Messaging

Table 7.TOE security functions provided by the platform

7.1.1 SF_CARD_AUTHENTICATION: Card authentication

This security function ensures the management of the authentication:

- The terminal is authenticated through the Card Manager's authentication mechanism, based on a one-time cryptographic challenge-response protocol.
- The Card Manager is the only card user authorized to open a secure channel.

As the user PIN is managed by the application itself, this security function manages the Card Manager's authentication that opens secure channel for communication with the terminal. Authentication failure is managed using a retry counter. When the predefined number of unsuccessful authentication is reached the card will be BLOCKED.

7.1.2 SF_CARD_CRYPTO: Card cryptographic algorithm and keys managements

This security function provides the cryptographic algorithm and functions used by the TSF:

- AES algorithm supports 128-bit key. The HW implementation of AES is used.
- TDES algorithm only supports 112-bit key and 168-bit key
- RSA algorithm supports 1024-to-2048 bits keys. The RSA algorithm is SW-based and does not use the IC cryptograhic library. The platform supports both standard and CRT RSA.
- Random generator uses the certified Hardware Random Generator that fulfils the requirements of AIS31 (see [ST/NXP]).
- SHA-1 and SHA-256 algorithms
- ECDH and ECDSA algortithms support 224, 256, and 384 bits keys. These ECC-based algorithms are implemented by the SW.

This security function controls all the operations relative to the card keys management.

Key generation: The TOE provides the following:

  - RSA key generation manages 1024 to 2048-bits long keys. The RSA key generation is SW and does not use the IC cryptographic library.
  - The TDES key generation (for session keys) uses the random generator.
- Key destruction: the TOE provides a specified cryptographic key destruction method that makes the content of the key unavailable.

This security function ensures the confidentiality of keys during manipulation and ensures the de-allocation of memory after use.

This security function is supported by the the IC security function **F.RNG (Random Number Generator), F.HW_DES (TDES Co-processor)** and **F.HW_AES (AES Co-processor)**.

7.1.3 SF_CARD_EMANATION: Emanation protection

This SF protects the cryptographic keys (D.ISD_KEYS, D.APP_KEYS) and the PINs (D.PIN) against snooping. The SF ensures that:

- The TOE shall not emit electromagnetic radiation in excess of unintelligible emission enabling access to the cryptographic keys and the PINs.
- The TOE shall ensure that the attacker S.OFFCARD is not able to use I/O, VCC or Ground interface to gain access to the cryptographic keys and the PINs.

This security function is supported by the IC security function **F.LOG (Logical Protection)**.

7.1.4 SF_CARD_INTEGRITY: Card objects integrity

This security function provides a means to check the integrity of data stored in EEPROM: the cryptographic keys (D.ISD_KEYS and D.APP_KEYS), the PINs (D.PIN), and the life cycle state (D.OP_REGISTRY).

In case of integrity error detection, this SF will prohibit the use of the altered data, and take appropriate actions: mute or terminate the card.

This SF also ensures that no residual information is available after a PIN deallocation.

This security function supports SF_CARD_PROTECT by checking platform data integrity before use.

This security function also provides the authorized users with the capability to verify the integrity of stored TSF executable code.

7.1.5 SF_CARD_MGR: Card manager

This security function ensures the administration of the card during its life-cycle: personalization phase, and usage phase. This SF enforces the following access control policies:

- Applet installation, and deletion
- Java objects access control (firewall)

This security function analyzes the incoming commands and checks the access rights, according to the life-cycle and the required secure environment.

This security function ensures that only authorized administrator can manage card contents and manages the following access control policies:

At the end of the platform initialization, the Card Manager (Issuer) security domain is created and the associated Card Manager keys are loaded before the Card Life Cycle state is set to OP-READY. Once in OP-READY state, the card is under Card Manager control.
Once the platform is set to OP-READY, applets can be installed by the authenticated Card Manager, through a successfully opened secure channel. Application security domains are created and the associated keys are loaded. Only an authenticated Card Manager is allowed to modify the card life-cycle, and update the keys. Access to Java objects is controlled by the firewall using the security context attached to each objects.
During usage phase, the Card Manager controls access to a Java object through the Firewall, using the Security context associated with each object.
The selection of an application is always allowed.

This security function is supported by SF_CARD_AUTHENTICATION and SF_CARD_SECURE_MESSAGING.

7.1.6 SF_CARD_PROTECT: Card operation protection

This security function ensures the protection of the TSF and supports the following operations.

Analyze potential violation on the card life-cycle inconsistency, the PIN and keys integrity error, the illegal access to Java objects, and the unavailability of resources.
Take action upon violation detection: reset the card, block the action, terminate or mute the card.
Resist to physical attacks (such as out-of-bound voltage, clock frequency and temperature, etc)

In case of error detections this function returns an error or an exception and takes appropriate shield action. If during the TSF execution an unexpected error or an abortion occurs, a secure state will be preserved by resetting security attributes to secure values and if necessary recover the persistently stored data to a secure consistent state. To this end, this security function ensures the atomicity of Java objects update in EEPROM as follows:

- The content of the data that are modified within a transaction is copied in the transaction dedicated EEPROM area. The TSF manages an optimistic backup: the optimistic backup mechanism includes a backup of the previous data value at first data modification, and previous value restoring at abort.
- Commit operation closes the transaction, and de-activates the dedicated transaction area.
- Rollback operation restores the original values of the objects (modified during the transaction) and de-activates the dedicated transaction area.
- The EEPROM containing sensitive data is kept in a consistent state whatever the time when EEPROM programming sequence stops, either during copying, invalidating, restoring data to or from the backup dedicated EEPROM area or updating sensitive data in EEPROM.

This security function is supported by the IC security functions **F.OPC (Control of Operating Conditions)** and **F.PHY (Protection against Physical Manipulation)** for attack detecting and resisting.

7.1.7 SF_CARD_ SECURE_MESSAGING: Card secure messaging

This security function ensures the integrity and/or the confidentiality of command/message transmission in a secure channel. The integrity is achieved by adding a message authentication code to the message. The confidentiality is achieved by APDU message data field encryption. These features are used in accordance with the security mode applied to the secure channel.

This security function is activated after a Card Manager authentication that allows the open of the secure channel. This security function ensures that the secure channel is closed after a select application command or in case of error detected within the session.

The secure channel is required for the applet Installation and CardManager keys (D.ISD_KEYS) loading.

This security funtcion is supported by SF_CARD_AUTHENTICATION.

8 Composition

Table 8 provides the dependencies of the current TOE security functions on the IC security functions.

IC Security functions vs. **TOE Security functions**	SF_CARD_AUTHENTICATION	SF_CARD_CRYPTO	SF_CARD_EMANATION	SF_CARD_INTEGRITY	SF_CARD_MGR	SF_CARD_PROTECT	SF_CARD_SECURE_MESSAGING
F.RNG		X					
F.HW_DES		X					
F.HW_AES		X					
F.OPC						X	
F.PHY						X	
F.LOG			X				

Table 8 Dependencies of TOE security functions on IC security functions

END OF SECURITY TARGET