IDMotion V2 Platform

Common Criteria / ISO 15408
Security Target – Public version
EAL5+

CONTENT

FIGURES

TABLES

1. REFERENCES

1.1 EXTERNAL REFERENCES

[ISO]	**ISO references**
[ISO7816]	Identification cards – Integrated circuit(s) cards with contacts - Books 1 to 9
[ISO14443]	Identification cards — Contactless integrated circuit(s) cards — Proximity cards — Books 1 to 4
[CC]	**Common Criteria references**
[CC-1]	Common Criteria for Information Technology Security Evaluation Part 1: Introduction and general model, CCMB-2006-09-001, version 3.1 rev 5, April 2017
[CC-2]	Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, CCMB-2007-09-002, version 3.1 rev 5, April 2017
[CC-3]	Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, CCMB-2007-09-003, version 3.1 rev 5, April 2017
[CEM]	Common Methodology for Information Technology Security Evaluation Methodology CCMB-2007-09-004, version 3.1 rev 5, April 2017
[ST-IC]	Security Target Common Criteria H13 IFX_CCI_000003h, IFX_CCI_000005h, IFX_CCI_000008h, IFX_CCI_00000Ch, IFX_CCI_000013h, IFX_CCI_000014h, IFX_CCI_000015h, IFX_CCI_00001Ch, , IFX_CCI_00001Dh Revision: 0.5 – 2017-05-22
[CR-IC]	*Certification Report,* BSI-DSZ-CC-0945-2017
[CRYPTO]	**Cryptographic references**

1.2 INTERNAL REFERENCES

[ALC-DVS]	ALC DVS Identification of security measures document Ref: D1445073
[MULTOS_ENABLEMENT]	MULTOS Enablement Reference: MAO-DOC-TEC-101 v1.2
[MULTOS_GLDA]	MULTOS Guide to Loading and Deleting Reference: MAO-DOC-TEC-008 v2.28
[MULTOS_MDRM]	MULTOS Developer's Reference Manual Reference: MAO-DOC-TEC-006 v1.54
[MULTOS_SGAD]	MULTOS Security Guidance for MULTOS Application Developers Reference: MI-MA-0031 v1.6
[MULTOS_GALU]	MULTOS Guide to Generating Application Load Units Reference: MAO-DOC-TEC-009 v2.9
[MULTOS_MVP]	MULTOS Mask Verification Procedure Reference: MI-PR-0012 v1.1

2. INTRODUCTION

2.1 SECURITY TARGET REFERENCE

Title :	IDMotion V2 Platform Security target
Version :	1.1
ST Reference :	ST_D1172991_P
Origin :	Gemalto
Author :	Franck OHAYON
IT Security Evaluation scheme :	Thales
IT Security Certification scheme :	Agence Nationale de la Sécurité des Systèmes d'Information (ANSSI)

Common Criteria: Version 3.1, Revision 5, April 2017
Identity of the Target of Evaluation (TOE): The Target of Evaluation is IDMotion V2 with OS MULTOS V4.5.2 platform mask on IFX_CCI_000014 family component (Infineon).

2.2 TOE REFERENCE

Product Name :	**IDMotion V2**
TOE Name :	IDMotionV2 Multos Platform
TOE Technical Product Reference*:	T1036817
Security Controllers:	IFX_CCI_000014 (Infineon)
TOE documentation :	AGD Documentation: [MULTOS_ENABLEMENT], [MULTOS_GLDA], [MULTOS_MDRM], [MULTOS_SGAD], [MULTOS_GALU], [MULTOS_MVP]

* Note: PDM reference
The Product and TOE identification details are provided in §Product description.

2.3 SECURITY TARGET IDENTIFICATION

Version of the TOE:
Each mask reference is identified by having an:

- OS type / OS version: 000452

Store in the NVM (constant defined in the source code)

- Build number: To be updated at the VLR

Stored in NVM. The build number is generated automatically by our build server and inserted into buildnumber.h before automatically building the source. This build number is used as a label in the Configuration Management System.

- Chip identity data (only byte 1 of the Chip Identification Data corresponding to the Platform Identifier): 0x16
- AMD Version Information : 01510001

Held in NVM. AMD version for IDMotionV2 is 0151v001 and has been allocated by MAOSCO and is a constant defined in the source code.

Get Configuration Data APDU Command:

CLA	INS	P1	P2	Lc	Data	Le
80	10	00	00	-	-	00

Get Configuration Data APDU Response

Token	Request	Data Returned
0x00 00	**Platform Identification**	**OS_type + OS_version**
0x01 00	Largest ALU Possible	max_alu_size
0x02 00	Communication Transfer Parameters	comms_tx_parameters
0x03 00	ATR Control	cold_reset_application_id + warm_reset_application_id
0x04 00	**AMD Version Information**	**amd_version_data**
0x05 00	Codelets available	codelet_list
0x06 00	Applications loaded	{application_id + application_memory_allocated}
0x07 00	MKD_PKC	MULTOS_pk_certificate
0x08 00	Codelet checksums	The 4 byte MULTOS checksum for each codelet listed in the same orders as the codelets in token 0x0500
0x09 00	ATS Control	application_ATS_selected.application_id or MULTOS_aid
0x0A 00	**Build Number**	**The build number of the implementation. Encoding defined by the implementer.**
0x0B 00	Primitives Supported	A list of bits, one bit per possible primitive (4 sets of 256 primitives, with a "1" indicating that the primitive is supported. The first byte contains the bits for set 0 primitives 0-7 (held in bits 0-7), the second byte for set 0 primitives 8-17 and so on.
0x0C 00	**Chip Identity Data**	**Silicon manufacturer specific chip identity data.**

2.4 SECURITY TARGET OVERVIEW

The integrated circuit card (ICC), or smartcard (or inlays/booklets, modules, or chips), is an ideal tool for the delivery of distributed, secure information processing at low cost. However, an application developed for one smartcard is usually not portable to another. Furthermore, many current smartcard operating systems allow only one application per card, meaning end users must carry a multitude of cards, one for each function or service required. Multos International, in its role as a member of the MULTOS Consortium (also known as MAOSCO), is developing an open, high-security multi-application operating system to address the current shortcomings of smartcard operating systems. This operating system is called MULTOS.
In order to satisfy the objectives set for it, MULTOS should be able to:
Execute an application written for MULTOS - application execution should be independent of the underlying smartcard hardware.
Load many applications - applications should be able to co-exist on the smartcard.
Ensure that applications are securely loaded and segregated - they should not be able to interfere with each other or with MULTOS.
In summary, MULTOS provides a common development and operating platform for smartcard applications. It allows multiple applications to be loaded onto a single smartcard and execute without interfering with or being interfered with by other applications. It also allows applications written for MULTOS to execute on different types of smartcard independent of the underlying smartcard hardware

2.5 TARGET OF EVALUATION DESCRIPTION

This part of the Security Target describes the Target of Evaluation as an aid to the understanding of its security requirements and addresses the product type, the intended usage and the general IT features of the TOE.

2.5.1 Product Description

MULTOS is an operating system for integrated circuit cards (also known as smartcards). It is designed to allow multiple smartcard applications to be securely loaded and executed on a smartcard.

The security is based on crypto libraries developed by Gemalto

Beside the TOE, the product also contain native applications and native modules used by ICAO application (out of scope of the TOE)

Mel applications and native modules:

- NATIVE/MEL Application: GMF v1.0
  The GMF application was initially developed to provide a global privacy protocol (PACE) protecting the access to all applications embedded on the ID Motion product.
  The GMF (Global Master File) represents – as its name says – the card's Master File and its sub-structures, files and keys. From a terminal point of view, it is an application that processes all commands performed under the Master File:
  - Selection/read of transparent EF (e.g EF.CardAccess for PACE)
  - Global Authentication commands for PACE
  - Selection of an application and re-routing of the commands to the appropriate selected application (this is transparent to the terminal)
- MEL Application: Pin Server Application (PSA) v0.2 [CMD GetData with Tag 0x0101]
  Unlike Global Platform based smartcards, MULTOS does not have a "global PIN" or Cardholder Verification Method (CVM). In the JavaCard/Global Platform world, the global CVM allows several applications to share a common, global Personal Identification Number (later PIN). This means that in a multi-applicative context, if the CVM has been verified by an application (e.g. to unlock an access condition), it does not need to be verified by all other applications that rely on the CVM validation
  As this feature is required by some of the ID Motion applications, namely IAS Classic, the service is implemented by the PSA application. In addition to managing the "standard" CVM, the PSA is in charge also of managing biometric CVMs.
- Native Application: ETravel EAC/PACE/BAC v2.4
  Native ICAO passport application embedded as a part of an ID Motion platform, based on MULTOS technology

- MEL Application: IAS Classic v4.4.1C [CMD GetData with Tag 0xDF30]
  IAS is an application that provides all the necessary functions to integrate a smart card in a public key infrastructure (PKI) system, suitable for identity and corporate security applications. It is also useful for storing information about the cardholder and any sensitive data. IAS implements state–of–the–art security and conforms to the latest standards for smart cards and PKI applications. It is also fully compliant with digital signature law.
- MEL Application: MOC client (MOCC) v1.0.2A [CMD GetVersion]
  MOC Client relies on PSA (PIN Server Application). The biometric templates are all stored within PSA and services are provided through MULTOS delegation

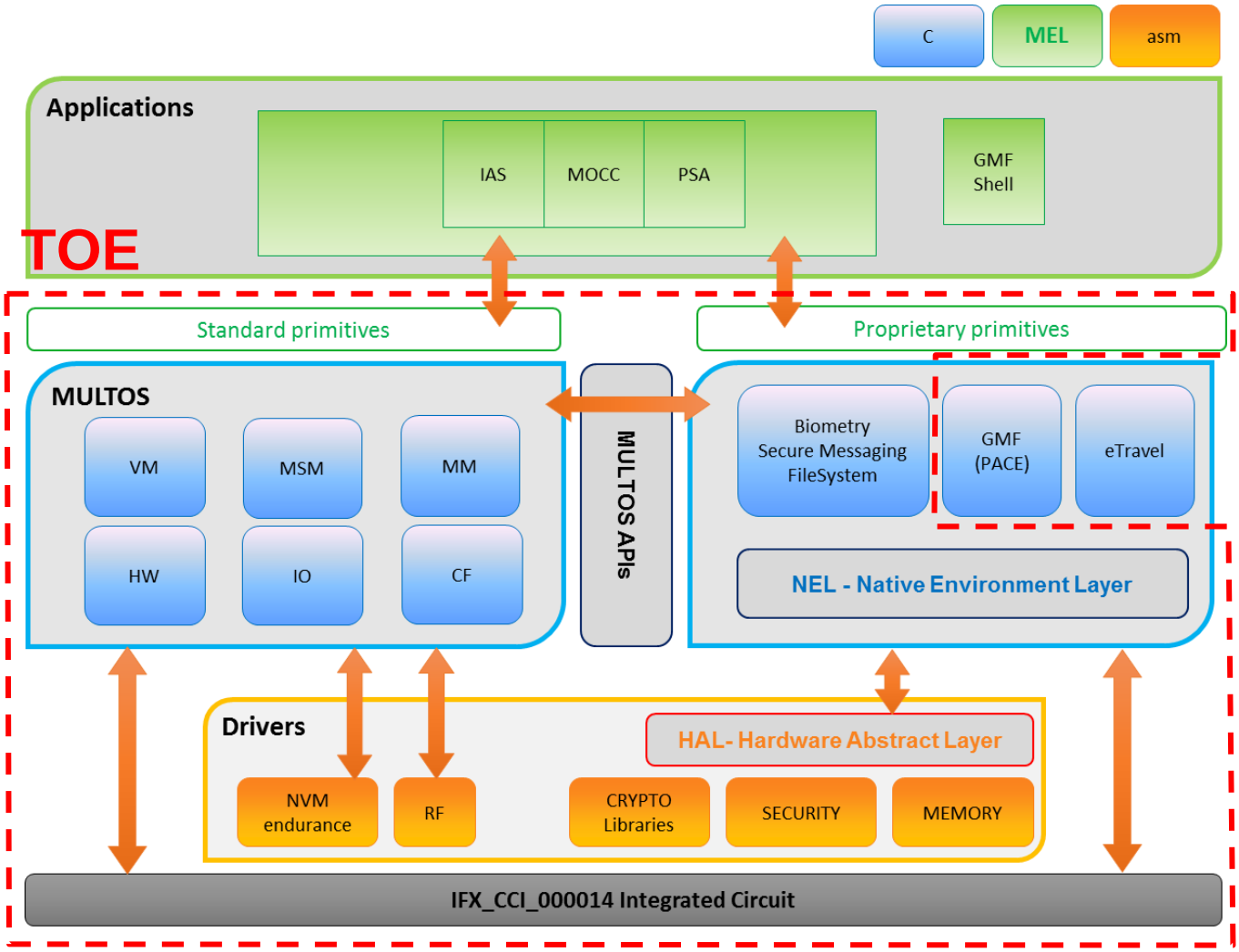

Figure 1: Layered Structure of MULTOS Software

VM: *Virtual Machine* ***CF:*** *Cryptographic Functions subsystem* ***NVM:*** *Non Volatile Memory*
MSM: *Multos Security Manager* ***IO:*** *I/O Communications subsystem*
MM: *Application Memory Manager Subsystem* ***HW:*** *Hardware Services subsystem*

The Biometry Secure Messaging regroup in fact 4 native blocs:
- ***MBIO:*** Fingerprint Match On Card algorithm
- ***MSM:*** ISO Secure Messaging services (Set & Clear session, Wrap & Unwrap)
- ***MFS:*** This module provides each ID Motion application with the ability to construct and manipulate its own file system in the NVM static area allocated to it by MULTOS.

- ***GMF:*** Native Commands (MGAM) – The GMF (Global Master File) sub-system represents – as its name says – the card's Master File and its sub-structures, files and keys. From a terminal point of view, it is an application that processes all commands performed under the Master File:
  - Selection/read of transparent EF (e.g EF.CardAccess for PACE)
  - Global Authentication commands (e.g PACE authentication)

The MBIO, MSM and MFS are not evaluated as TSF for this evaluation.

The user of the smartcard accesses the applications loaded on the MULTOS operating system via an Interface Device (IFD), which could be a Point-of-Sale terminal, Automatic Teller Machine, or some other device which supports ISO 7816 smartcard protocols.
Communications across the IFD-MULTOS interface comprise a message transmitted by the smartcard when it is reset (the Answer-to-Reset or ATR message), followed by command-response pairs, where a command is a message from the IFD to MULTOS and a response is a message from MULTOS to the IFD.
By means of these command-response pairs, MULTOS allows:

a) Applications to be loaded into and deleted from the smartcard.
b) An IFD to access data and applications which are loaded on the card.
c) Information specific to the card to be retrieved by an IFD.

MULTOS is a single-threaded operating system. Only one application can be executing at any given time. MULTOS does not provide mechanisms for concurrency or multi-tasking. Following power-on of the smartcard and initialization, the basic execution sequence for MULTOS is as follows:

a) Wait for input from the IFD.
b) Parse the input.
c) If the input is a MULTOS command, process the command and write a response to the IFD.
d) Otherwise, execute the currently selected application and write to the IFD any output created by the application.
e) Loop back to a).

Applications to be loaded on MULTOS-based smartcards are written in a hardware-independent language called MULTOS Executable Language (MEL). MEL applications are interpreted by MULTOS, rather than being compiled and executed directly on the smartcard processor.
MULTOS also provides for shared code routines, called Codelets, which can be called by an executing application. Codelets can be loaded into MULTOS during IC manufacture or at smartcard personalization time. A codelet has its own code address space but executes in the context of the calling application, so has access to the application's data.
MULTOS is targeted to operate on the Infineon Technologies IFX_CCI_000014 family Smartcard Integrated Circuits (ICs). The IC provides the microprocessor to execute the instructions comprising the executable code of MULTOS.
The Infineon Technologies IFX_CCI_000014 family is a dual interface integrated circuit see Hardware reference manual for details [HW-Manual].

2.5.2 Intended Method of Use

MULTOS is intended to provide a hardware-independent environment for the execution of multiple applications that provide a variety of functions and services to the holder of the smartcard. Applications may be developed and supplied by different organizations from different industries, and consequently may provide many different services e.g., financial, communication or access control. The security requirements of different applications may also vary (i.e., some applications may require a high level of security while others may only have a low level or no security requirements).
A user of a MULTOS-equipped smartcard will be able to select any of the loaded applications and execute them. The user will access the facilities of the smartcard via an appropriate IFD. MULTOS implements a command interface for handling commands received from the IFD.
MULTOS provides a number of system calls (called primitives) which allow the currently executing application to request particular services from MULTOS.
MULTOS provides the following features:

- MULTOS will ensure all requests to load applications are appropriately authorized. MULTOS will support a capability to ensure the authenticity and integrity of an application when loading the application onto the smartcard. MULTOS will also ensure all requests to delete applications are appropriately authorized. Reasons for wishing to delete applications may be because they are found to contain errors, because an updated application is available, or to make room on the smartcard for a more desirable application.

- MULTOS will support a capability to load encrypted applications onto the smartcard, decrypt such applications and make them available to the smartcard user for execution

- MULTOS will ensure no application loaded on the smartcard can interfere with the operation of any other loaded application or with MULTOS. MULTOS will also ensure that an application's code and data will not be available to other applications after it has been deleted. MULTOS will provide the capability to authenticate a card as a valid MULTOS equipped smartcard.

- MULTOS will provide the capability to restrict the use of regulated features of the smartcard (e.g., strong cryptography) to authorized applications.

- MULTOS defines certain functions (installing keys, loading applications and deleting applications) as sensitive functions. For each of these functions, if the number of failed attempts to execute the function reaches a pre-defined limit over the life of the smartcard, MULTOS will permanently disable the function. In the case of installing keys, this means the card is unusable, as no applications can be loaded until keys have been installed. In the cases of application loading and deleting, other functions of the card remain available.

It is assumed that authorized applications which are loaded and executed by MULTOS are responsible for the secure processing of their own information. MULTOS provides an environment for secure loading and execution of smartcard applications.
The MULTOS access control policy maintains separate storage and execution space for applications loaded into an MCD (MULTOS Carrier Device). The application execution management mechanism ensures each application, including its code and data areas, is kept separate from every other application loaded on the MCD. Each application that is restricted to its own code and data space cannot gain access to the code or data of another loaded application.

2.5.3 Actors

Actors	Identification
Integrated Circuit (IC) Developer	IFX
Embedded Software Developer	Gemalto (See [ALC-DVS] for details)
Integrated Circuit (IC) Manufacturer	IFX
Initializer	Gemalto (See [ALC-DVS] for details)
Pre-personalizer	Gemalto (See [ALC-DVS] for details)
Personalization Agent	The agent who is acting on the behalf of the issuing State or Organization and personalize the MRTD (Machin Readable Travel Document) for the holder by activities establishing the identity of the holder with biographic data.
Card Holder	The rightful holder of the card for whom the issuer personalizes it.

Table 1: Identification of the actors

2.5.4 Smartcard Product Life Cycle

The Smartcard product life-cycle is decomposed into seven phases, according to the "Smartcard Integrated Circuit Protection Profile".

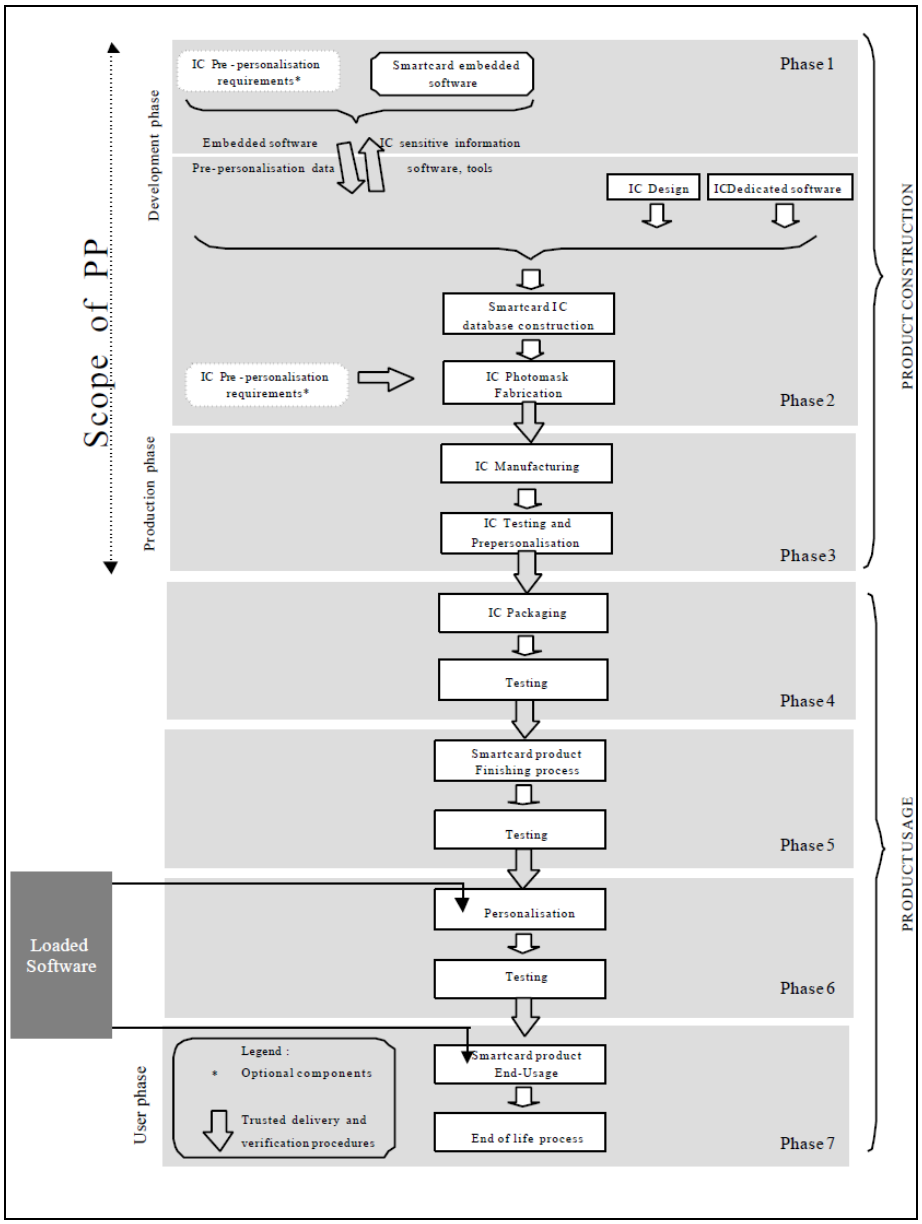

Figure 2: Smartcard IC with Multi-Application Platform Life Cycle

The product and TOE life cycle phases are described in table 1. The table also mentions the actor(s) involved in each phase, as well as the associated location(s).

The IC does not contain any part of the IDMotion V2 software prior to phase 5. The loading of the IDMotion V2 software occurs during phase 5 in the flash memory, after which the IC Platform software loading service is locked and no more available (No patching process is possible after phase 5).
As described, at the end of phase 5 Gemalto delivers platform personalized IDMotion V2 product to our customers. At this stage, the TOE is entirely built and protects itself through the security mechanisms implemented in the operating system and the underlying IC. Consequently, the TOE delivery point - which determines the boundary between the ALC and AGD Common Criteria assurance classes - is put at the end of phase 5.

Notes related to Gemalto applications development

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

Note related to patch development

No patch is present within the TOE for the present evaluation. The patch mechanism is disabled at the end of the phase 5 (pre-personalization)

Evaluation scope: life-cycle boundary

Phase	Description / comments		Who	Where
1	IDMotion V2 platform development	Platform development, Primitives integration & tests	**Gemalto** MULTOS R&D team secure environment	**Gemalto Sydney** **Gemalto Fareham UK**
			Gemalto SL Crypto team secure environment	**Gemalto Meudon (Fr)** **Gemalto La Ciotat (Fr)**
			Gemalto GBU R&D team secure environment	**Gemalto Meudon** **Gemalto Singapore**
2	IC development	IC development	**IC developer** secure environment	IC development site(s) Refer to [CR-IC]
3	IC manufacturing	Manufacturing of virgin integrated circuits embedding a flash loader protected by a dedicated transport key.	**IC manufacturer** secure environment	IC manufacturing site(s) Refer to [CR-IC]
4	SC manufacturing: IC packaging, also called “assembly”	IC packaging & testing	**Gemalto or IFX (for contactless only)** Production teams secure environment	**Gemalto Gémenos** **Gemalto Singapore** **Gemalto Vantaa**
5	SC manufacturing & pre-personalization	▪ Module embedding (ICC, smartcard, inlays/booklets, modules, or chips) ▪ Loading of the Gemalto software (platform and applications) ▪ SC initialization (profile building, loading of data needed for card pre-personalization…)	**Gemalto** Production teams secure environment	**Gemalto Gémenos** **Gemalto Singapore** **Gemalto Vantaa** **Gemalto Tczew** **Gemalto Curitiba** **Gemalto Montgomery**
6	SC Personalization	Creation of files and loading of end-user data	**SC Personalizer:** Gemalto or another accredited company secure environment	SC Personalizer site
7	End-usage	End-usage for SC issuer	**SC Issuer**	Field
		End-usage for cardholder	**Cardholder**	Field

Table 2: Product and TOE life-cycle phases

Remark1: Initialization & pre-personalization operation could be done on module or on other form factor. The form factor does not affect the TOE security.

Remark2: Alternative life cycle, wafer are shipped by Infineon to form factor manufacturer (no module manufacturing required) and initialization /pre-personalization is done in Gemalto site.

Remark3: For initialization/pre-personalization IC flash loader could be used based the IC manufacturer recommendation.

Remark4: Embedding (module put on a dedicated form factor) will be done on an audited site defined in the phase 5 of the *"Table 2: Product and TOE life-cycle phases"*

Remark5: The Phase 5 correspond to the IDMotion V2 Issuance process describe as below for the parts called "Flash Loading/Card Manufacturing" and "Card Pre-enablement".

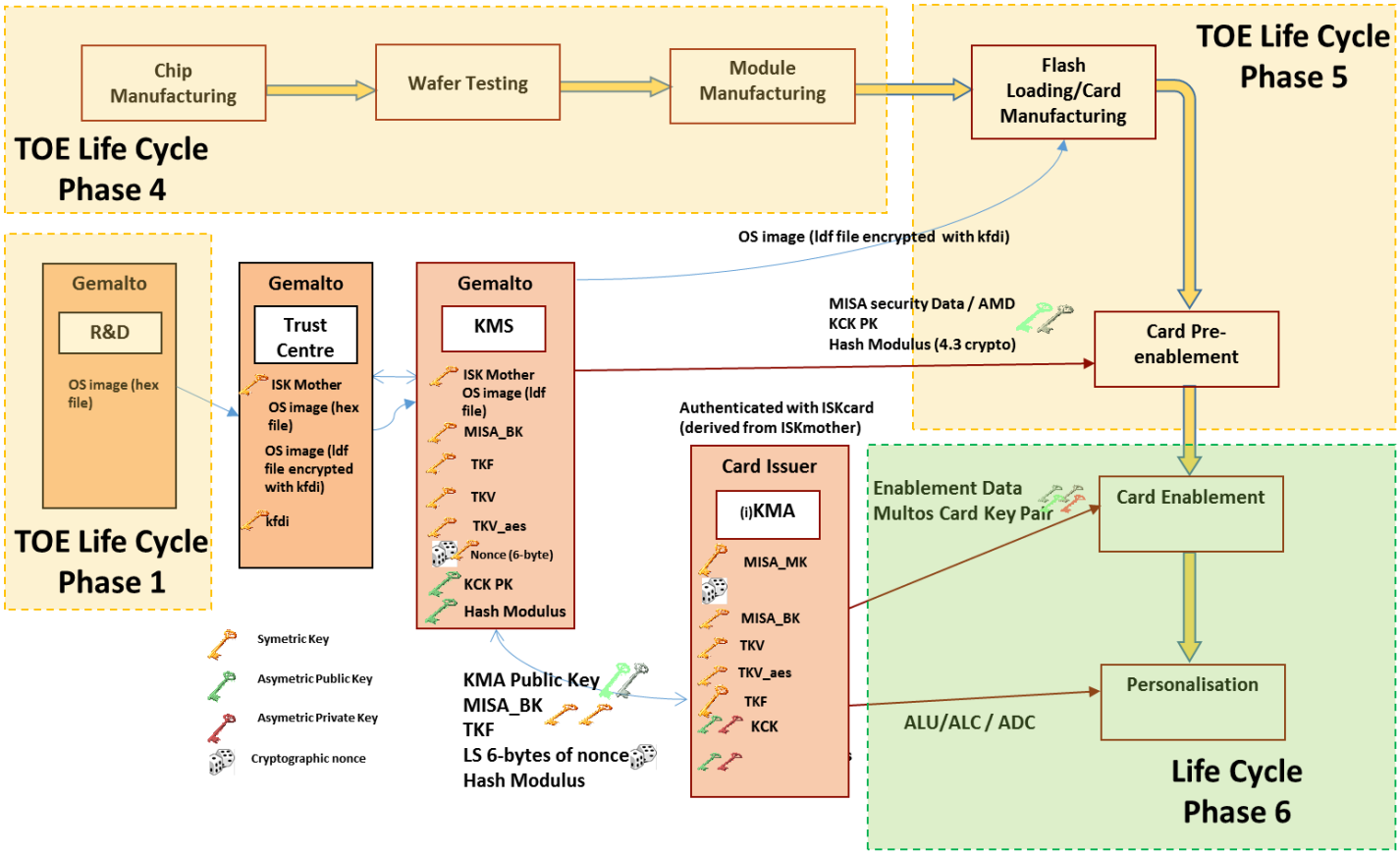

Figure 3: MULTOS Platform Life Cycle

2.5.5 Target of Evaluation Location and Usage

MULTOS will initially be developed in software. Following successful implementation and testing, the MULTOS executable will be masked in Flash Memory and embedded on smartcards, inlays/booklets, modules, or chips.
Once the MULTOS chip has been embedded, interaction with it will be via commands issued to smartcards (or inlays/booklets, modules, or chips) from an IFD or service requests (i.e., MULTOS system calls, known as primitives) made by an executing application.

2.5.6 Supporting Firmware

MULTOS requires firmware run-time libraries and RF libraries to support writing data to flash memory. These libraries are supplied by Infineon Technologies. They provide low-level routines to support writing data to flash memory, which is used on the target smartcard for the storage of applications. MULTOS requires the run-time libraries to execute correctly according to specification, to ensure data is written to the correct address within flash memory.

2.5.7 Supporting Security Infrastructure

It is assumed MULTOS-equipped smartcards and MULTOS applications will be manufactured and distributed within a commercial framework providing a procedural security infrastructure. Figure 3 provides a simplified context diagram of the MULTOS commercial framework and security infrastructure.

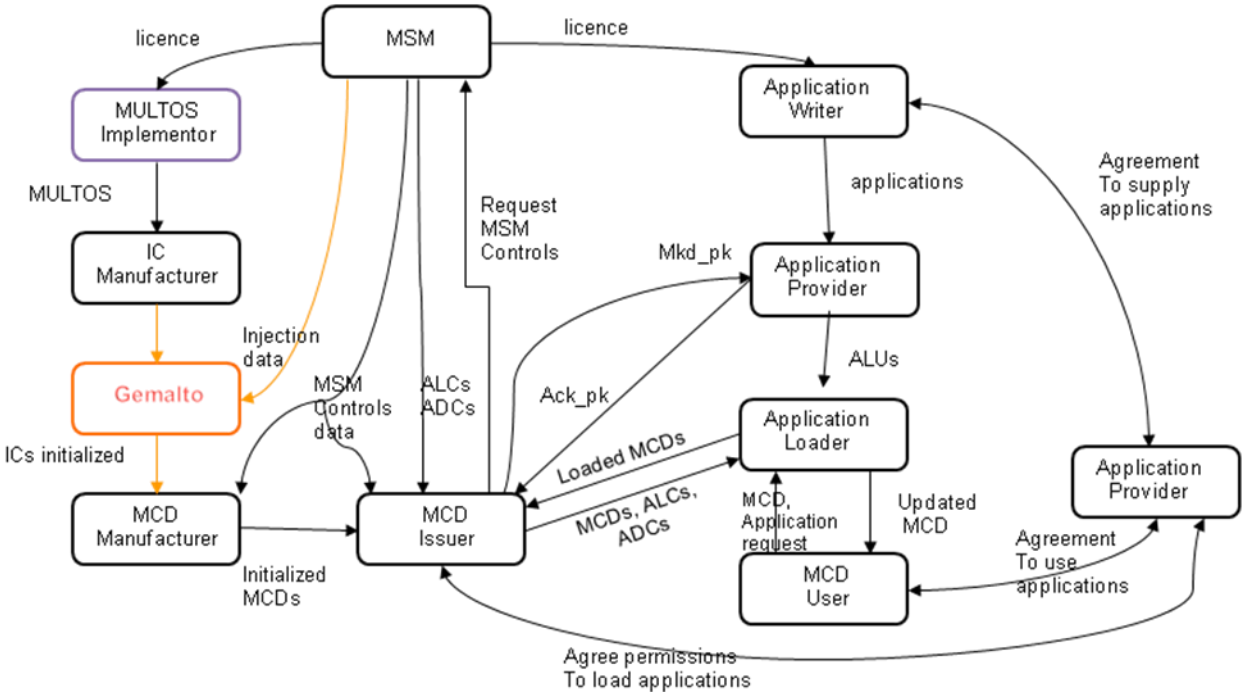

Figure 4: MULTOS Infrastructure Context Diagram

Legend

MSM: *Multos Security Manager*
MCD: *Multos Carrier Device*
ALU: *Application Load Unit*
ALC: *Application Load Certificate*
ADC: *Application Delete Certificate*
Mkd_pk: *Held by MSM; copy provided to Application Providers; used by Application Providers to encrypt KTU for target MCD.*
Ack_pk: *Provided to MCD Issuer, who gets it certified by the MSM when ALCs and ADCs are requested*

In Figure 3, the box labeled “MSM” includes MAOSCO in addition to the MSM role.

In this product version including Gemalto the role of the IC Manufacturer is split.

- The founder (Infineon) manufactures the chips and performs limited FLASH Memory injection including a diversified transport key (used by Initialization Mode) and supplies them in flash download mode.
- Gemalto then performs the final manufacturing step by FLASH downloading the MULTOS OS and injecting the iKMA keys and data into the FLASH memory.

The following roles and responsibilities are assumed within the infrastructure:

a) **MULTOS Security Manager (MSM):** defines and polices the MULTOS security infrastructure and provides criteria and services necessary for MULTOS participants to operate within the infrastructure. It acts as Certification Authority for the security infrastructure. It is assumed only one MSM exists. The MSM must be trusted by all participants in the infrastructure

b) **MULTOS Implementor:** the organization that implements a MULTOS version.
The MULTOS Implementor is licensed by MAOSCO and provides its MULTOS version to the IC Manufacturer. The MCD Issuer requests the MSM to provide MSM Controls Data, although this may be delivered to the MCD Manufacturer or MCD Issuer.

c) **Integrated Circuit (IC) Manufacturer:** manufacturer of silicon from which chips and smartcards are made. It is assumed the IC Manufacturer is trusted to perform its tasks correctly: This includes:

- To perform limited Flash memory injection, and supplies the chip in flash downloading mode.
- The injection of diversified transport key (used by Initialization Mode).

d) **Gemalto:** is included in the new scheme in order to perform the final manufacturing step by injecting the iKMA keys and data into the Flash memory;

The initialized ICs are provided to MCD Manufacturers:

a) **MULTOS Carrier Device (MCD) Manufacturer:** responsible for embedding the IC in its plastic carrier and for background printing on the card. The result is an initialized MCD. This operation is assumed not to be security sensitive. The MCD Manufacturer may also receive MSM Controls Data from the MSM and enable the MCDs. Initialized and enabled MCDs are provided to MCD Issuers.

b) **MCD Issuer:** responsible for issuing to users the MCD itself. The MCD Issuer may also enable initialized MCDs, by loading MSM Controls Data received from the MSM onto the MCDs. MCD Issuers retain the ultimate authority over what applications are loaded on their MCDs. MCD Issuers register applications with the MSM, provide information related to the applications and receive application load and delete certificates from the MSM.

c) **Application Writer:** licensed by MAOSCO to produce applications for MULTOS. Supplies applications under contract to Application Issuers.

d) **Application Issuer:** an organization that wishes to offer an application to MCD Users.
The Application Issuer agrees with an MCD Issuer that the application can be loaded onto MCDs belonging to the MCD Issuer.

e) **Application Provider:** the organization that takes responsibility for an application, by certifying it with the organization’s public key and encrypting it where necessary. The Application Provider is a role that can be performed by an Application Writer, Application Issuer or MCD Issuer, rather than necessarily, being an organization in its own right.

f) **Application Loader:** responsible for performing the technical operation of loading applications onto MCDs. The Application Loader enters into an agreement with one or more Application Issuers and MCD Issuers for loading applications supplied by one or more Application Providers.

g) **MCD User:** final user of the MCD.

The MSM authorizes potential MULTOS platforms. To receive MSM authorization, a platform must comply with criteria covering attributes of the platform itself and the procedures associated with its manufacture.
MULTOS platforms are assumed to satisfy the following requirements:

a) They are manufactured in a controlled environment conforming to MSM rules.

b) They are subject to type approval by the MSM.

c) They possess a level of tamper resistance.

2.5.8 Application Load Units (ALU)

An Application Load Unit (ALU) is generated by an Application Provider to load applications. An ALU may be uncertified or certified. An uncertified ALU simply contains a clear text copy of the application. A certified ALU contains, in addition to the application, an application signature, which authenticates the application. The Application Provider may also encrypt parts of the application, in which case a Key Transformation Unit is included in the certified ALU.

2.5.9 Key Transformation Unit (KTU)

An Application Provider wishing to utilize application confidentiality will generate a Key Transformation Unit (KTU). The KTU contains descriptors for the areas of the application's code and data that have been encrypted. Each descriptor contains the start address of the protected area, the length of the protected area, an indicator of the algorithm used and the key used to encrypt the contents of the area. The descriptors and some header information (including application identifier and target MCD number) are then encrypted, using the target MCD's public transport key, and included in the KTU.

2.5.10 Application Load and Delete Certificates (ALCs & ADCs)

Application Load Certificates (ALCs) and Application Delete Certificates (ADCs) are generated by the MSM to respectively load and delete an application on to and from an MCD. Each ALC contains the unique Application ID of the application for which it is created. Each ALC refers to a particular domain, which defines the set of MCDs that the application may be loaded on to and deleted from.
The domain is defined by a set of load permissions and may be:

a) A specific MCD.
b) A subset of the cards issued by an MCD Issuer.
c) All cards issued by an MCD Issuer.
d) Limited to a subset of cards enabled on specific dates.
e) A combination of the above.

An ALC contains load controls that define exactly what load operations are allowed.
The load controls specify:

a) If application signature has been used.
b) If application confidentiality has been used.
c) If reloading a deleted application is permitted.

The ALC also contains feature permissions, which define what regulated features the application may use:

- Access flags for access to MULTOS functions:
    - Cryptographic access
    - Allow access to particular interfaces (contact/contactless)
    - Allow access to shared PIN
    - Allow access to card block/unblock primitives
    - …
- Application Permissions - used to match ALC with ALU & Card, or ADC to Card – plus one-time load (history)

The ADC for an application is created at the same time as the ALC. It contains the same unique Application ID and the same set of load permissions as the corresponding ALC.

2.5.11 Keys

The following table lists each of the cryptographic keys required to support the MULTOS security infrastructure. Each key is identified by a name, the key type (symmetric/asymmetric) and its role within the MULTOS security infrastructure. Asymmetric keys have two components: a secret and a public key. In the following table, secret components of asymmetric keys are identified by a "_sk" suffix, and public components by the suffix "_pk".

Key name	Key definition	Key part	Role
MISA_mk *symmetric*	MISA Master Key		Generated by MSM; used by MSM to generate MISA_bk
MISA_bk *symmetric*	MISA Base Key		Each key value is unique to a given MISA. Used by MISA and MSM to determine TKV for a specific MCD These keys are diversified during wafer production to inject each chip with a unique transport key (TKV)
TKV *symmetric*	MCD-specific transport key		Generated by MSM; stored in FLASH memory of target MCD; used by MSM to encrypt MCD-specific MSM Controls Data and also by MULTOS to decrypt the MSM Controls Data. Unique key used to encrypt enablement data
TKF *symmetric*	Fixed part of MCD-specific transport key		Generated by MSM; stored in FLASH memory of MCD; used by MSM, MCD Issuer and Application Loader to check authenticity of target MCD; TKF is fixed for all MCDs
MKD *asymmetric*	MCD-specific asymmetric transport key.	mkd_sk	Held in FLASH memory of target MCD; used by MULTOS to decrypt KTU (Key Translation Unit)
		mkd_pk	Held by MSM; stored in FLASH memory of target MCD; copy provided to Application Providers; used by Application Providers to encrypt KTU (Key Translation Unit) for target MCD
		mkd_pk_c	mkd_pk, certified by MSM using tkck_sk to indicate its authenticity. By decrypting this with tkck_pk the mkd_pk can be recovered for use
TKCK *asymmetric*	Transport Key Certification Key	tkck_sk	Held securely by MSM; used by MSM to certify MCD-specific public transport keys (mkd_pk) Used to sign the mkd_pk during enablement data generation
		tkck_pk	Held by MSM; copy provided to Application Providers; used by Application Providers to verify and retrieve certified MCD-specific public transport keys (mkd_pk_c)
DEK *symmetric*	Data Encryption Key		Used to encipher sections of the application. It is included as part of the KTU, which is secured by the chip's key (MKD).
ACK *asymmetric*	Application Provider's asymmetric key *Generated by Application Provider*	ack_sk	Held by Application Provider; used by Application Provider to sign application certificate
		ack_pk	Provided to MCD Issuer, who gets it certified by the MSM when ALCs and ADCs are requested
HM *asymmetric*	Hash Modulus		While not strictly a key as such, this RSA public key is used as an input the MULTOS proprietary Asymmetric Hash algorithm which is based on RSA. This is used during the verification of ALC/ADCs, MSM controls and application signatures The secret part of the key is not use
KCK *asymmetric*	Global Key Certification Key *Generate by MSM*	kck_sk	Held securely by MSM; used by MSM to certify ADCs and ALCs (and indirectly through these, Application Provider public keys (ack_pk)); Guarantees the authenticity of applications
		kck_pk	Held in FLASH memory of every MCD ; used by MULTOS to verify ALCs, ADCs and Application Provider public keys

Table 3: MULTOS Security Infrastructure Keys

The critical keys, which are managed by the MSM and support the MULTOS security infrastructure, are:

a) Global Key Certification Key (KCK)
b) Transport Key Certification Key (TKCK)

The KCK supports the authentication of MULTOS applications and the authorization of requests by MCD Issuers to load and delete applications. The secret KCK (kck_sk) is held securely by the MSM and is used to sign ALCs and ADCs. ALCs and ADCs contain the Application Provider's public key (ack_pk), so signing the ALC/ADC also certifies ack_pk for use with MCDs. The public KCK (kck_pk) is installed in the Flash memory of each instance of MULTOS (i.e., it is available on every MCD).

The TKCK supports the provision of application confidentiality and MCD authentication. An asymmetric transport key is created for each MCD (this is MKD). The public part of MKD (mkd_pk) is certified by the MSM using the secret part of the TKCK (tkck_sk). Application Providers wishing to utilize application confidentiality when loading applications onto an MCD obtain from the MCD Issuer the public part of MKD, certified by the MSM (i.e., mkd_pk_c). The Application Provider uses the public part of the TKCK (tkck_pk) to authenticate mkd_pk and uses mkd_pk to encrypt the KTU for the target MCD.

2.5.12 MULTOS Initialization Security Data (Card Pre-enablement)

The MISA (MULTOS Injection Security Application) Security Data contains the unique identity and transport keys used to identify and protect the device during distribution. The security of a MULTOS device and the MULTOS scheme depends upon this data.

MULTOS Initialization Security Data is generated by the MSM and supplied to Gemalto for incorporation into MULTOS. MULTOS Initialization Security Data comprises the security data, which are injected.

The MULTOS security data is injected during MULTOS initialization.
This is performed using a device called a MULTOS Injection Security Application (MISA).
The MSM constructs data for each MISA, including a unique MISA identifier.

Two crypto schemes can be used:
- 2 keys multi-key DES to protect a MULTOS device.
- AES to protect devices during distribution.
It will be necessary for a manufactured device to either use the DES or AES crypto scheme and for this to be identified and decided at manufacturing.

The MISA then constructs the data to be injected into the MCD (table 3).

Name	Definition	Description
security_data	jump_code	Internal use
	random_number	
	msm_controls_algorithm_id	Determines whether the security_data_des_key or security_data_aes_key structure is used and present in the MISA security data structure.
	[security_data_des_key] or [security_data_aes_key]	
[security_data_des_key]	tkf	MCD-specific symmetric transport keys (TKF and TKV), which are used in loading the MCD-specific asymmetric transport key (MKD) as a component of the MSM Controls Data
	tkv	
	mism_id	Two bytes ID that identify the keyset used by the card Keyset used to generate the TKV
	icc_serial_number	A unique identifier based on the MISA identifier and ICC serial number
	initialisation_date	Initialization date, indicating when the security data was injected into the MCD
	security_level	A security flag indicating MSM Controls Data has not been loaded. (Redundancy usage with OS)
	random_number	
[security_data_aes_key]	tkf	MCD-specific symmetric transport keys (TKF and TKV), which are used in loading the MCD-specific asymmetric transport key (MKD) as a component of the MSM Controls Data
	tkv_aes	
	tkv_aes_length	
	mism_id	Two bytes ID that identify the keyset used by the card Keyset used to generate the TKV
	icc_serial_number	A unique identifier based on the MISA identifier and ICC serial number
	initialisation_date	Initialization date, indicating when the security data was injected into the MCD
	security_level	A security flag indicating MSM Controls Data has not been loaded. (Redundancy usage with OS)
	random_number	

Table 4: MISA security Data

Figure 4 depicts the information flow from the MSM to the Gemalto.

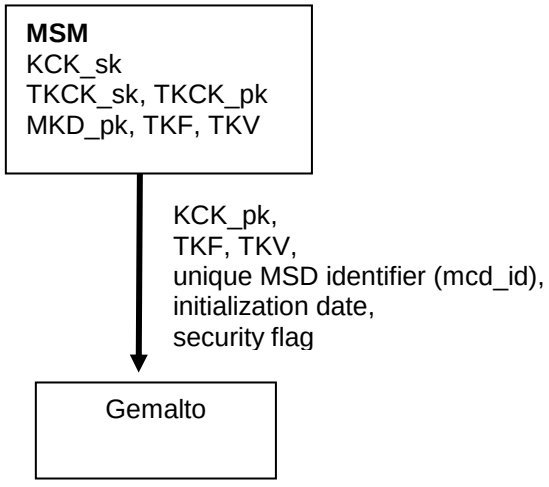

Figure 5: MULTOS Initialization Security Data Information Flow

2.5.13 MSM Controls Data (Card enablement)

The MCD must be loaded with its permissions and asymmetric transport key set (i.e., MKD) before application loading can be supported. The transport key (comprising the private key, and certified public key) and permissions are provided by the MSM to the MCD Manufacturer or MCD Issuer in MSM Controls Data. This data also includes the MCD's unique identifier and is protected by the MCD-specific symmetric transport key (TKV). Once the transport keys have been generated and encrypted, MSM destroys the copy of mkd_sk it generated, in order to ensure the confidentiality of this key.
Figure 5 depicts the information flow from the MSM to the MCD Manufacturer or MCD Issuer.

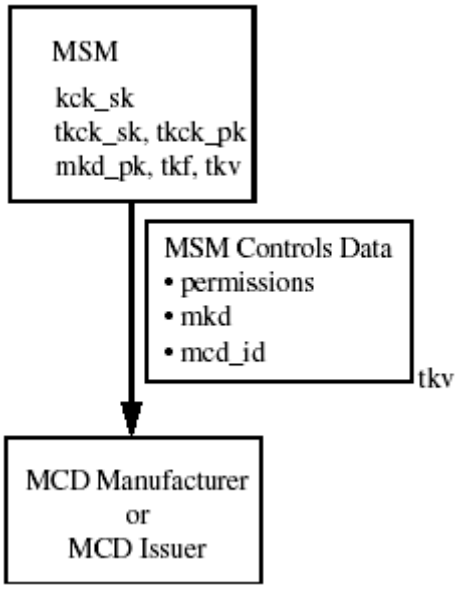

Figure 6: MSM Controls Data Information Flow

2.5.14 Loading Applications (Personalization)

The principal key and data exchanges involved in loading applications onto MCDs are depicted in Figure 5.

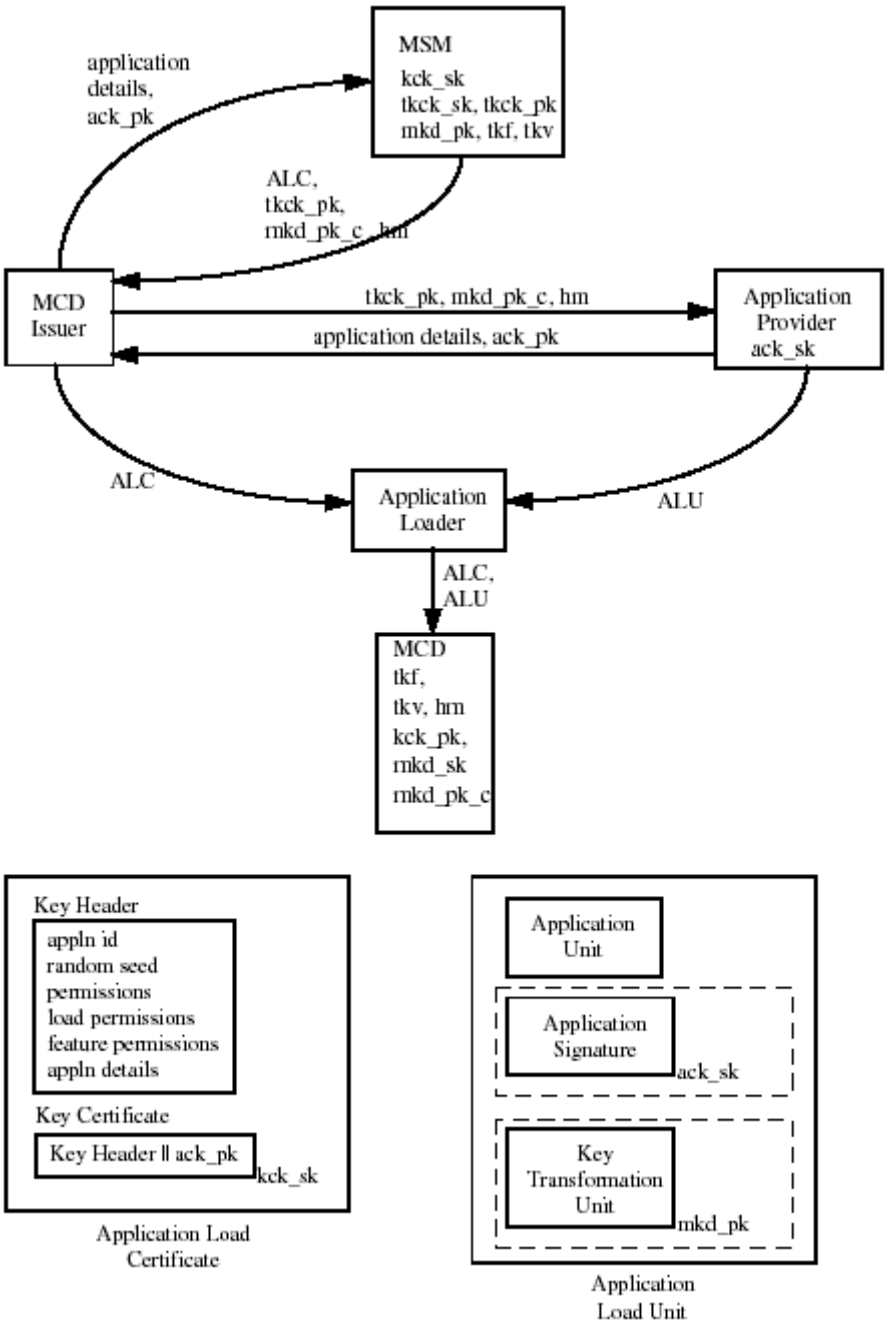

Figure 7 : Principal Key and Data Exchanges in Loading MCD Applications

The Application Provider provides its public key (ack_pk) and details of the application to be loaded to the MCD Issuer.
The MCD Issuer forwards ack_pk and the application details to the MSM.
The MSM creates an ALC, which contains the application details in the Key Header.
MSM creates the Key Certificate over the information in the Key Header, concatenated with ack_pk, and signs it with the secret KCK (kck_sk).
The MSM provides the ALC to the MCD Issuer. If the Application Provider has requested use of application confidentiality, the MSM also provides the MCD Issuer with the target MCD's certified public transport key (mkd_pk_c) and the public TKCK (tkck_pk).
The MCD Issuer provides mkd_pk_c and tkck_pk to the Application Provider and the ALC to the Application Loader.
The Application Provider creates an ALU for the application to be loaded onto the MCD.
The ALU comprises the following components:

a) Application unit, containing the application's code and data.
b) Application signature (optional).
c) KTU (optional).

If the Application Provider requires application authentication, it includes an application signature in the ALU. The application signature is created over the application unit and signed with the Application Provider's secret key (ack_sk).
If the Application Provider requires application confidentiality, it includes a KTU. The KTU is signed using the target MCD's public key (mkd_pk), retrieved from mkd_pk_c using tkck_pk.
The Application Provider provides the ALU to the Application Loader. The Application Loader loads the ALU on the target MCD, using the ALC to demonstrate the load has been authorized by the MSM.

3. CONFORMANCE CLAIMS

This section describes how the ST claims conformance with Common Criteria for Information Technology Security Evaluation Version 3.1, Revision 5.

3.1 COMMON CRITERIA CONFORMANCE CLAIMS

This ST has been built with Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 5, as the following:

- Part 2 conformant.
- Part 3 conformant with EAL5 level augmented.

The EAL5 level from CC Part 3 is augmented with the assurance components ALC_DVS.2 and AVA_VAN.5.

3.2 PROTECTION PROFILE CLAIM AND PACKAGE CLAIM

This ST is based on PP/0010 Version 2.0, Issue November 2000, registered at the French Certification Body. Please note that the PP/0010 is upwardly compatible with the PP/9806 and PP/9911. Therefore, this ST is also based on Smartcard IC Protection Profile PP/9806, Version 2.0, Issue September 1998 and Smartcard IC with Embedded Software Protection Profile PP/9911, Version 2.0, Issue June 1999.

Note: Items which are common to PP/9806 and PP/0010 are indicated by a "*" in this Security Target.

4. SECURITY PROBLEM DEFINITION

This section describes the security problem to be addressed by the TOE and the operational environment in which the TOE is intended to be used. It provides a description of the assets to be protected, the threats, the organizational security policies and the assumptions about the operational environment of the TOE.

4.1 ASSETS

Assets are security relevant elements of the TOE that include:
Assets linked to the IC with Multi-Application Secure Platform itself:

- The IC specifications, design, development tools.
- The IC Dedicated software.
- The integrity of the Multi-Application Platform Software.
- Multi-Application Platform specifications, implementation, test programs and related documentation.
- The confidential TSF data (TKF, TKV and mkd_sk).

Assets are also linked to Loaded-Applications on the platform:

- Application provider User Data:

Loaded-Application software loaded on the platform. Confidential Loaded-Application SF data. (Encrypted SF data for the eventual Loaded Application Security Functions).

- The TOE resources:
  - Card resources: memory space and computation power made available to a Loaded-Application and its security functions.

Assets are also linked to end user, card holder and application provider:

- End User Data for users of Native Applications.
- End User Data for users of Loaded Applications.

NOTE: even if the PP scope does not include the applications, the TOE must provide security mechanisms such that Loaded Applications can protect the End User data when required.
Assets have to be protected in terms of confidentiality, authenticity and control of their origin.

4.2 THREATS

The TOE and its operational environment as defined in chapter 2 are required to counter the threats described hereafter.
A threat agent (an attacker) wishes to abuse the assets either by functional attacks or by environmental manipulation, by specific hardware manipulation, by a combination of hardware and software manipulations or by any other type of attacks.
Threats have to be split in:

- Threats against which specific protection within the TOE is required (class I).
- Threats against which specific protection within the environment is required (class II).

4.2.1 Unauthorized Full or Partial Cloning of the Target of Evaluation

T.CLON*
Functional cloning by attackers of the TOE (full or partial) appears to be relevant to all phases of the TOE life-cycle, from phase 1 to phase 7, but only phases 1 and 4 to 7 are considered here, since functional cloning in phases 2 and 3 are purely in the scope of Smartcard IC PP. Generally, this threat is derived from specific threats by attackers, addressing User Data and potentially TSF data, combining unauthorized disclosure, modification or theft of assets at different phases.

4.2.2 Threats on Phase 1

Common Criteria v3 does not require threats for the development environment so these threats (and any references to them) should be ignored for this version of Common Criteria. Instead, Common Criteria v3 requires that the development environment is evaluated in the ALC assurance class of the evaluation.

During phase 1, three types of threats by attackers have to be considered:

a) Threats on the Smartcard Embedded Software (ES) and its development environment, such as unauthorized disclosure, modification or theft of the Smartcard Embedded Software and/or initialization data.

b) Threats on the assets transmitted from the IC designer to the Smartcard software developer during the Smartcard ES development.

c) Threats on the Smartcard Embedded Software and initialization data transmitted during the delivery process from the Smartcard software developer to the IC designer.

Unauthorized disclosure of assets
This type of threat covers unauthorized disclosure of assets by attackers who may possess a wide range of technical skills, resources and motivation. Such attackers may also have technical awareness of the product.

T.DIS_INFO* (type b)
An attacker may cause unauthorized disclosure of the assets delivered by the IC designer to the Smartcard Embedded Software developer, such as sensitive information on IC specification, design and technology, software and tools if applicable.

T.DIS_DEL* (type c)
An attacker may cause unauthorized disclosure of the Asset Smartcard Embedded Software and any additional *application data* (such as IC pre-personalization requirements) during the delivery to the IC designer.
NOTE application data means TSF data.

T.DIS_ES1 (type a)
An attacker may cause unauthorized disclosure of ES (technical or detailed specifications, implementation code) and/or TSF data (such as secrets, or control parameters for protection system, specification and implementation for security mechanisms).

T.DIS_TEST_ES (type a and c)
An attacker may cause unauthorized disclosure of the Smartcard ES test programs or any related information.

Theft or unauthorized use of assets
Potential attackers may gain access to the TOE and perform operations for which they are not authorized. For example, such an attacker may personalize, modify or influence the product in order to gain access to the Smartcard application system.

T.T_DEL* (type c)
An attacker may target theft of the Smartcard Embedded Software and any additional *application data* (such as pre-personalization requirements) during the delivery process to the IC designer.
NOTE application data means TSF data.

T.T_TOOLS (type a and b)

An attacker may target theft or unauthorized use of the Smartcard ES development tools (such as PC, development software, databases).

T.T_SAMPLE2 (type a)
An attacker may target theft or unauthorized use of TOE samples (e.g. bond-out chips with the Embedded Software).

Unauthorized modification of assets
The TOE may be subjected by attackers to different types of logical or physical attacks, which may compromise security. Due to the intended usage of the TOE (the TOE environment may be hostile), the TOE security may be bypassed or compromised reducing the integrity of the TOE security mechanisms and disabling their ability to manage the TOE security.
This type of threats includes the implementation of malicious Trojan horses.

T_MOD_DEL* (type c)
An attacker may cause unauthorized modification of the Smartcard Embedded Software and any additional *application data* (such as IC pre-personalization requirements) during the delivery process to the IC designer.
Note: Application data means TSF data.

T.MOD (type a)
An attacker may cause unauthorized modification of ES and/or TSF data or any related information (technical specifications).

4.2.3 Threats on Delivery for/from Phase 1 to Phases 4 to 6

Threats by attackers on data transmitted during the delivery process from the Smartcard developer to the IC packaging manufacturer, the Finishing process manufacturer or the Personalizer.
These threats are described hereafter:

T.DIS_DEL1
An attacker may cause unauthorized disclosure of and ES personalization Data during delivery to the IC Packaging manufacturer, the Finishing process manufacturer or the Personalizer.

T.DIS_DEL2
An attacker may cause unauthorized disclosure of ES personalization Data delivered to the IC Packaging manufacturer, the Finishing process manufacturer or the Personalizer

T.MOD_DEL1
An attacker may cause unauthorized modification of ES personalization Data during delivery to the IC Packaging manufacturer, the Finishing process manufacturer or the Personalizer.

T.MOD_DEL2
An attacker may cause unauthorized modification of and ES personalization Data delivered to the IC Packaging manufacturer, the Finishing process manufacturer or the Personalizer.

4.2.4 Threats on Phases 4 to 7

During these phases, the assumed threats could be described in four types:

- Unauthorized disclosure of assets.
- Theft or unauthorized use of assets.
- Unauthorized modification of assets.
- Threats on Loaded-Applications.

Unauthorized disclosure of assets
This type of threat covers unauthorized disclosure of assets by attackers who may possess a wide range of technical skills, resources and motivation. Such attackers may also have technical awareness of the product.

T. DIS_ES2
An attacker may cause unauthorized disclosure of ES, Native-Application, and Loaded-Application TSF Data (such as data protection system, memory partitioning, cryptographic programs and keys).

Theft or unauthorized use of assets
Potential attackers may gain access to the TOE and perform operation for which they are not allowed. For example, such attackers may personalize the product in an unauthorized manner, or try to gain fraudulently access to the Smartcard system.

T.T_ES
An attacker may cause unauthorized use of TOE. (e.g. bond out chips with embedded software).

T.T_CMD
An attacker may cause unauthorized use of instructions or commands or sequence of commands sent to the TOE.

Unauthorized modification of assets
The TOE may be subjected by attackers to different types of logical or physical attacks, which may compromise security. Due to the intended usage of the TOE (the TOE environment may be hostile), the TOE security parts may be bypassed or compromised reducing the integrity of the TOE security mechanisms and disabling their ability to manage the TOE security. This type of threat includes the implementation of malicious Trojan horses, Trapdoors, downloading of viruses or unauthorized programs.

T.MOD_TSF
An attacker may cause unauthorized modification or destruction of TOE Security Function Data. (By any mean including probing, electronic perturbation etc.)

T.MOD_LOAD
An attacker may cause unauthorized loading of Native Applications. This includes also illegal modification of eventual Native Applications. As the TOE described in a Security Target claiming this PP must include eventual Native Applications, their loading or modification must be blocked during the usage phase. The threat includes bypassing this blocking.

T.MOD_EXE
An attacker may cause unauthorized execution of Platform or application software.

T.MOD_SHARE
An attacker may cause unauthorized modification of Platform or application behavior by interaction of different programs.

T.MOD_SOFT*
An attacker may cause unauthorized modification of Smartcard Embedded Software and data.

4.2.5 Threats on Phases 6 to 7

Threats on assets linked to Loaded-Applications
These threats by attackers are specific to the Multi-Application Platform. They are centered on threats by attackers to loading/unloading of Loaded-Applications and to threats using a Loaded-Application to attack another.

T.LOAD_MAN
Attackers loading an application on the platform bypassing the Administrator. This threat could lead to undue usage of card resources, and for unverified application to attack on other Loaded-Application TSF or User data.

T.LOAD_APP
Attackers loading an application that purports to be another Loaded-Application. This attacks card resources and end user data.

T.LOAD_OTHER
Attackers loading the software representation of a Loaded-Application intended for a specific platform domain onto other platform domains, thus taking from the Loaded-Application representation the security feature of being confined to a specific domain. This is an attack on Loaded-Application User Data.

T.LOAD_MOD
Attackers intercepting application load units and altering code or data without the permission of the Loaded-Application Provider. This attacks application provider user data.

T.APP_DISC
Attackers intercepting application load units and gaining access to confidential code or data. This is an attack on application provider user data's confidentiality and knowledge.

T.APP_CORR
Attackers loading an application that partially or completely overwrites other Loaded-Applications, either corrupting or gaining access to code or data. This is an attack on Application Provider user data.

T.APP_REMOVE
Attackers removing a Loaded -Application without the involvement of the Administrator. This is an attack on Application Provider user data.

T.ERR_REMOVE
Attackers removing a Loaded-Application leaving confidential data and/or code in memory which can be examined This is an attack on Application Provider user data.

T.DEL_REMOVE
Attackers removing a Loaded-Application at the same time deleting part or all of another Loaded-Application. This is an attack on Application Provider user data.

T.APP_READ
Attackers using a loaded application to read confidential data or code belonging to another Loaded-Application. This attacks the confidentiality of User Data.

T.APP_MOD
Attackers using a Loaded-Application to modify data or code belonging to another Loaded-Application without its authorization. This is an attack on Application Provider user data (and also End User data).

T.RESOURCES
Attackers targeting total or partial destruction of card resources delivered by the platform.
NOTE: T.APP_DISC is also present during phase A1.

4.2.6 Threats on Phase 7

Unauthorized disclosure of assets
T.DIS_DATA
Attackers may cause unauthorized disclosure of User (application provider and end user) data and TSF data.

Unauthorized modification of assets
T.MOD_DATA
Attackers may cause unauthorized modification or destruction of User (application provider and end user) Data and TSF data.
Table 2 given below indicates the relationship between the phases of the Smartcard life cycle, the threats and the type of the threats:

Threats	Phase 1	Phase A1	Phase 4	Phase 5	Phase 6	Phase 7
T.CLON*	Class II		Class I	Class I	Class I	Class I
T.DIS_INFO*	Class II					
T.DIS_DEL*	Class II					
T.DIS_DEL1	Class II		Class II	Class II	Class II	
T.DIS_DEL2			Class II	Class II	Class II	
T.DIS_ES1	Class II					
T.DIS_TEST_ES	Class II					
T.DIS_ES2			Class I	Class I	Class I	Class I
T.T_DEL*	Class II					
T.T_TOOLS	Class II					
T.T_SAMPLE2	Class II					
T.T_ES			Class I	Class I	Class I	Class I
T.T_CMD			Class I	Class I	Class I	Class I
T.MOD_DEL*	Class II					
T.MOD_DEL1	Class II		Class II	Class II	Class II	
T.MOD_DEL2			Class II	Class II	Class II	
T.MOD	Class II					
T.MOD_TSF			Class I	Class I	Class I	Class I
T.MOD_SOFT*			Class I	Class I	Class I	Class I
T.MOD_LOAD			Class I	Class I	Class I	Class I
T.MOD_EXE			Class I	Class I	Class I	Class I
T.MOD_SHARE			Class I	Class I	Class I	Class I
T.DIS_DATA						Class I
T.MOD_DATA						Class I
T.LOAD_MAN					Class I	Class I
T.LOAD_APP					Class I	Class I
T.LOAD_OTHER					Class I	Class I
T.LOAD_MOD					Class I/II	Class I/II
T.APP_DISC		Class II			Class I/II	Class I/II
T.APP_CORR					Class I	Class I
T.APP_REMOVE					Class I	Class I
T.ERR_REMOVE					Class I	Class I
T.DEL_REMOVE					Class I	Class I
T.APP_READ					Class I	Class I
T.APP_MOD					Class I	Class I
T.RESOURCES					Class I	Class I

Table 5: Relationship between phases and threats

Note: Phases 2 and 3 are covered in the scope of Smartcard IC PP.

4.3 ORGANIZATIONAL SECURITY POLICIES

OSP.CIPHER The TOE must contribute and provide cryptographic functions are required to actually protect the exchanged information. These cryptographic algorithms need to be consistent with cryptographic usage policies and standards.
Remark that even if the TOE shall provide access to the appropriate TSFs, it is still the responsibility of the Applications to use them.

OSP.CONF-ALU The confidential ALU includes a KTU (key transformation unit) used to encrypt sensitive sections of the ALU (or the entire ALU). KTU itself is encrypted off-card using the card's public asymmetric transport key (MKD-PK). MULTOS decrypts it using its private asymmetric transport key (MKD-SK).

4.4 ASSUMPTIONS

Security always concerns the whole operational environment of the TOE. The weakest element of the chain determines the total system security. Assumptions described hereafter must be considered for a secure system using Smartcard products.

4.4.1 Assumptions on the Target of Evaluation Delivery Process (Phases 4 to 7)

Procedures shall guarantee the control of the TOE delivery and storage process and conformance to its objectives as described in the following assumptions:

A.DLV_PROTECT*
Procedures shall ensure protection of TOE material/information under delivery and storage.

A.DLV_AUDIT*
Procedures shall ensure that corrective actions are taken in case of improper operation in the delivery process and storage.

A.DLV_RESP*
Procedures shall ensure that people dealing with the procedure for delivery have got the required skill.

4.4.2 Assumptions on Phases 4 to 6

A.USE_TEST*
It is assumed that appropriate functionality testing of the TOE is used in phases 4, 5 and 6.

A.USE_PROD*
It is assumed that security procedures are used during all manufacturing and test operations through phases 4, 5, 6 to maintain confidentiality and integrity of the TOE and of its manufacturing and test data (to prevent any possible copy, modification, retention, theft or unauthorized use).

4.4.3 Assumption on Phase 7

A.USE_DIAG*
It is assumed that secure communication protocols and procedures are used between Smartcard and terminal.

4.4.4 Assumption on Loaded-Application Development (Phase A1)

A.APPLI_CONT
Whenever a Loaded-Application is to be loaded on the platform, it is assumed that its development and production follow the Administrator Guidance.

5. SECURITY OBJECTIVES

5.1 SECURITY OBJECTIVES FOR THE TARGET OF EVALUATION

The TOE shall achieve the following IT security objectives, and for that purpose, when IC physical security features are used, the specification of those IC physical security features shall be respected. When IC physical security features are not used, the Security Objectives shall be achieved in other ways:

O.TAMPER_ES
The TOE must prevent tampering with its security critical parts. In particular, the security mechanisms must prevent the unauthorized change of functional parameters, security attributes and secrets such as the life cycle sequence flags and cryptographic keys.

O.SIDE
The ES must be designed to avoid interpretations of electrical signals from the hardware part of the TOE.

O.CLON*
The TOE functionality must be protected from cloning.

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

O.FLAW*
The TOE must not contain flaws in design, implementation or operation.

O.DIS_MECHANISM2
The TOE shall ensure that the ES security mechanisms are protected against unauthorized disclosure.

O.DIS_MEMORY*
The TOE shall ensure that sensitive information stored in memories is protected against unauthorized disclosure.

NOTE sensitive information means *User Data and TSF data.*

O.MOD_MEMORY*
The TOE shall ensure that *sensitive information* stored in memories is protected against any corruption or unauthorized modification.
NOTE sensitive information means *User Data and TSF data.*
The following security objectives are necessary to meet the new threats specific to Multi-Application Platforms. This is why these objectives are new and not present in PP/9911.
.
O.ROLLBACK
The TOE must be in a well-defined valid state before the loading of an application, even in case of failure of the previous loading or removal. A failure must not hinder the resources that the TOE can deliver. A rollback operation can be achieved either through specific commands or automatically.

O.RESOURCE
The TOE must provide the means of controlling the use of resources by its users and subjects so as to prevent permanent unauthorized denial of service. (For example it must prevent a Loaded-Application from taking control of the whole permanent memory (FLASH) thus prohibiting other Loaded-Applications from using it).

O.LOAD
Loaded-Applications are only to be loaded onto a platform with the permission of the administrator.

O.SECURITY
The application load process must be able to guarantee, when required, the integrity, confidentiality, and to verify the claimed origin of the Loaded-Application code and data.
O.EFFECT_L

Loading an application must have no effect on the code and data of existing Loaded-Applications.

O.REMOVE
Removal of a Loaded-Application and consequent reuse of the Loaded-Application space is only to be performed with the authorization of the administrator. The space must not hold any information relative to data or code linked to the removed Loaded-Application.

O.EFFECT_R
Removal of a Loaded-Application must have no effect on the code and data of the remaining independent Loaded-Applications.

O.SEGREGATE
Loaded-Applications are to be segregated from other Loaded-Applications. A Loaded-Application may not read from or write to another Loaded-Application's code or data without its authorization.
Detailed information could be found in the MULTOS Architecture Specification - Application Abstract Machine [AAM] TEC-MAO-101-004/v4.3.1

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. Security Objectives for the Operational Environment

O.DECIPHER
MULTOS CARD decrypts the KTU

5.1.1 Objectives on Phase 1

Note that these objectives for phase 1 are described to maintain compatibility with PP/0010 which is compliant to Common Criteria v2.1. Common Criteria v3 does not require objectives for the development environment so these objectives (and any references to them) should be ignored for this version of Common Criteria. Instead, Common Criteria v3 requires that the development environment is evaluated in the ALC assurance class of the evaluation.

O.DEV_TOOLS*
The Smartcard ES shall be designed in a secure manner, by using exclusively software development tools (compilers assemblers, linkers, simulators, etc.) and software-hardware integration testing tools (emulators) that will result in the integrity of program and data.

O.DEV_DIS_ES
The Embedded Software developer shall use established procedures to control storage and usage of the classified development tools and documentation, suitable to maintain the integrity and the confidentiality of the assets of the TOE.
It must be ensured that tools are only delivered and accessible to the parties authorized personnel.
It must be ensured that confidential information on defined assets is only delivered to the parties' authorized personnel on a need-to-know basis.

O.SOFT_DLV*
The Embedded Software must be delivered from the Smartcard software developer (Phase I) to the IC designer through a trusted delivery and verification procedure that shall be able to maintain the integrity of the software and its confidentiality, *if applicable.*
NOTE: In PP00/10 it will be always considered applicable.

O.INIT_ACS
Initialization Data shall be accessible only by authorized personnel (physical, personnel, organizational, technical procedures).

O.SAMPLE_ACS
Samples used to run tests shall be accessible only by authorized personnel.

5.1.2 Objectives on the Target of Evaluation Delivery Process (Phases 4 to 7)

O.DLV_PROTECT*
Procedures shall ensure protection of TOE material/information under delivery including the following objectives:

- Non-disclosure of any security relevant information.
- Identification of the element under delivery.
- Meet confidentiality rules (confidentiality level, transmittal form, reception acknowledgement).
- Physical protection to prevent external damage.
- Secure storage and handling procedures (including rejected TOEs).
- Traceability of TOE during delivery including the following parameters:
- Origin and shipment details.
- Reception, reception acknowledgement.
- Location material/information.

O.DLV_AUDIT*
Procedures shall ensure that corrective actions are taken in case of improper operation in the delivery process (including if applicable any non-conformance to the confidentiality convention) and highlight all non-conformance to this process.
O.DLV_RESP*
Procedures shall ensure that people (shipping department, carrier, reception department) dealing with the procedure for delivery have got the required skill, training and knowledge to meet the procedure requirements and be able to act fully in accordance with the above expectations.

5.1.3 Objectives on Delivery from Phase 1 to Phases 4, 5 and 6

O.DLV_DATA
Native-Application and ES data must be delivered from the Smartcard embedded software developer (phase 1) either to the IC Packaging manufacturer, the Finishing Process manufacturer or the Personalizer through a trusted delivery and verification procedure that shall be able to maintain the integrity and confidentiality of the ES (Note: some application data are not required for embedding and are then delivered directly to phases 4 to 6).

5.1.4 Objectives on Phases 4 to 6

O.TEST_OPERATE*
Appropriate functionality testing of the TOE shall be used in phases 4 to 6.
During all manufacturing and test operations, security procedures shall be used through phases 4, 5 and 6 to maintain confidentiality and integrity of the TOE and its manufacturing and test data.

5.1.5 Objectives on Phase 7

O.SAMPLE_ACS Secure communication protocols and procedures shall be used between the Smartcard and the terminal.

5.1.6 Objectives on Loaded-Application Development and Loading (Phases A1 and A2)

This Objective is specific to Loaded-Application development in the Smartcard IC with Multi-Application Platform environment.
O.APPLI_DEV
The Loaded-Application provider must:

- Follow the Administrator Guidance and ensure that the applications are compliant with the Security Guidance for Application Developers. Amongst other topics application should ensure that sensitive assets are suitable protected (i.e encrypted)
- Provide trusted delivery channel so that the integrity and origin of the Loaded-Application can be verified and that its confidentiality can be maintained.

5.2 SECURITY OBJECTIVES RATIONALE

This section demonstrates that the stated specific security objectives address, and can be traced to, all security environment aspects identified. Each specific security objective being correlated to at least one threat, one OSP or one assumption.

5.2.1 Discussion of Threats, OSP and Security Objectives

The following discussion shows which security objectives counter which threats and enforce which OSP, phase by phase.
During phase 1, the Smartcard ES is being developed and the pre-personalization and personalization requirements are specified for all other phases.
The Target of Evaluation (TOE) is a functional product designed during phase 1, considering that the only purpose of the Embedded Software is to control and protect the operation of the Smartcard during phase 4 to 7 (operational phases). The global security requirements of the TOE mandate to consider, during the development phase, the security threats of the other phases. This is why the PP addresses the functions used in phases 4 to 7 but developed during phase 1. Then, the limit of the TOE corresponds to phase 1 including the TOE delivery to the IC manufacturer.

T.CLON*
The TOE being constructed can be cloned, but also the construction tools and document can help clone it. During phase 1, since the product does not exist, it cannot contribute to countering the threat. For the remaining phases 4 to 7, TOE participates in countering the threats.

T.DIS_INFO*
This threat addresses disclosure of specification, design and development tools concerning the IC and delivered to the software developer (during phase 1) in order to meet with the overall security objectives of the TOE. This threat is countered by development environment.

T.DIS_DEL*
This threat addresses disclosure of specifications, test programs, related documents, ES and data which is delivered from phase 1 to phase 2 for software embedding. As the TOE does not yet exist, the threat can only be countered by development environmental procedures.

T.DIS_DEL1
This threat addresses disclosure of software and ES Data during delivery from phase 1 to phases 4 to 6. As the data is not yet implemented in the TOE, the threat can only be countered by environmental procedures.

T.DIS_DEL2
This threat addresses disclosure of software or data which has been delivered, from phase 1, to phases 4 to 6. As the data is not yet implemented in the TOE, the threat can only be countered by environmental procedures.

T.DIS_ES1
The ES and accompanying documents are created and used during phase 1. As during this phase the product does not yet exist, it cannot contribute to countering the threat which must be countered by development environment.

T.DIS_ES2
Disclosure of ES and TSF data can compromise security. During phases 4 to 7, the TOE must counter the unauthorized disclosure of the ES and the Loaded-Application Data.

T.DIS_TEST_ES
Tests concerning the embedded software or software to be embedded is carried out in phase 1. This threat is countered by environmental development procedures, of which the tests themselves are part.

TT_DEL
The threat addresses the theft of software or ES Data which is delivered for software embedding, from phase 1 to phase 2. As the data is not yet implemented in the TOE, the threat can only be countered by developmental environmental procedures.

T.T_TOOLS
TOE development tools are used only during phase 1, so this threat can only exist during phase 1. As the TOE is not yet manufactured, this threat is countered by environmental procedures.

T.T_SAMPLE2
TOE samples are used only during phase 1, so this threat can only exist during phase 1. The theft or unauthorized use of samples are countered by environmental procedures.

T.MOD_DEL*
This threat addresses modification of software or TSF data which is delivered for software embedding, in phase 1. As the TOE does not exist during this phase, the threat must be countered by development procedures.

T.MOD_DEL1
This threat addresses modification of ES Personalization Data during delivery from embedded software developer, phase 1, to the IC packaging manufacturer, phase 4, the finishing process manufacturer, phase 5, and for the Personalizer, phase 6. As the data is not yet loaded on the TOE, the threat can only be countered by environmental procedures.

T.MOD_DEL2
This threat addresses modification of ES Personalization Data which is delivered to the IC packaging manufacturer, phase 4, the finishing process manufacturer, phase 5, and for the Personalizer, phase 6. As the data is not yet loaded on the TOE, the threat can only be countered by environmental procedures.

T.MOD
Modification of ES and TSF Data can be done during ES design in phase 1. Since the product does not exist, the threat can only be countered by environment procedures.

T.MOD_SOFT*

Once present on the TOE, the software and Application data can be modified in an unauthorized way during any phases from 4 to 7. This threat is countered by the TOE.

T.T_ES
This threat covers the unauthorized use of cards during the different phases of the card life cycle as well as the misappropriation of rights of Smartcards. This threat covers phases 4 to 7 and is countered by the TOE.

T.T_CMD
This threat includes the diversion of the hardware or the software, or both, in order to execute non authorized operations. This threat covers phases 4 to 7 and is countered by the TOE.

T.MOD_LOAD, T.MOD_EXE, T.MOD_SHARE
The loading of Native Applications, execution and modification of software can endanger the security of the TOE, and especially create interference between applications. This threat covers phases 4 to 7 and is countered by the TOE.

New threats not specific to Multi-Application platforms:

T.MOD_TSF
Modification of TOE Security function can only appear when the TOE exists, thus only during phases 4 to 7. This threat is countered by the TOE.

T.DIS_DATA
Threats on end user data can only appear after the data has been created, thus in usage phase 7. This threat is countered by the TOE.

T_MOD_DATA
Threats on end user data can only appear after the data has been created, thus in usage phase 7. This threat is countered by the TOE.

New Threats specific to Multi-Application platforms:

These threats are present during phases 6 to 7 depending when the Loaded-Applications are loaded. It can be supposed that Loaded-Applications are mostly used during phase 7.

T.LOAD_MAN
This threat comes from illegal loading of Loaded-Applications which can for example clone legal Loaded-Applications on other cards. Loading can be done in phases 6 and 7. This threat is countered by the TOE.

T.LOAD_APP
This threat is a complement to the precedent one. In this case, an illegal Loaded-Application is loaded in place of a legal one. The attacking party can be the same as above. This threat appears during phases 6 and 7 and is countered by the TOE.

T.LOAD_OTHER
This threat addresses loading a Loaded-Application to a domain to which it should not have access. This means that the other Loaded-Application can be attacked. This threat appears during phases 6 and 7 and is countered by the TOE.

T.LOAD_MOD
This threat alters code or data without the permission of the Loaded-Application Provider. This threat appears during phases 6 and 7 and is countered by the TOE and the environment.

T.APP_DISC
This is an attack on the Loaded-Application provider know how, and possibly on confidential data loaded along with the Loaded-Application. This threat appears during phases 6 and 7 and is countered by the TOE and the environment.

T.APP_CORR

This attack destroys partly or completely the other Loaded-Application, or more subtly can divert the Loaded-Application to create a dangerous state. This threat appears during phases 6 and 7 and is countered by the TOE.

T.APP_REMOVE
This threat addresses illegal removal of a legal Loaded-Application. It attacks reliability of services. This threat appears during phases 6 and 7 and is countered by the TOE.

T.ERR_REMOVE
This opportunistic threat takes advantage of a removal operation to attack the confidentiality of Loaded-Application provider know how, or confidential data. This threat appears during phases 6 and 7 and is countered by the TOE.

T.DEL_REMOVE
This threat is on remaining Loaded-Applications which can be damaged during the removal operation. This threat appears during phases 6 and 7 and is countered by the TOE.

T.APP_READ
This threat loads a Trojan horse to illegally access to confidential data belonging to other Loaded-Applications. This threat appears during phases 6 and 7 and is countered by the TOE.

T.APP_MOD
This threat loads a Trojan horse to illegally access to modify data or code belonging to another Loaded-Application. This threat appears during phases 6 and 7 and is countered by the TOE.

T.RESOURCES
This threat is aimed at the reliability of service of the platform or Loaded-Application. This threat appears during phases 6 and 7 and is countered by the TOE.
Threat T.APP_DISC is also present during Loaded-Application development, phase A1 when it is countered by the environment.

Organizational Security Policy:
None defined.

5.2.2 Threats & OSP Addressed by Security Objectives

5.2.2.1 Security objectives for the TOE

During phase 1, as the TOE does not yet exist, there is no threat/OSP on the TOE itself. For the phases 4 to 7, the following descriptions and tables indicate that each threat and OSP is mapped to at least one specific security objective during the life of the TOE:

Threats/OSP/Obj	O.TAMPER_ES	O.SIDE	O.OPERATE*	O.FLAW*	O.DIS_MECHANISM2	O.DIS.MEMORY*	O.MOD_MEMORY*	O.CLON*
T.CLON*					X	X		X
T.DIS.ES2		X	X	X	X	X		
T.T_ES	X		X	X			X	
T.T_CMD	X		X	X			X	
T.MOD_SOFT*	X		X	X			X	
T.MOD_LOAD	X		X	X			X	
T.MOD_EXE	X		X	X		X	X	
T.MOD_SHARE	X		X	X		X	X	
T.MOD_TSF	X		X	X			X	
T.DIS_DATA		X	X	X		X		
T.MOD_DATA	X		X	X			X	

Table 6: Mapping of security objectives to threats & OSP relative to phases 4 to 7

The TOE shall use state of the art technology to achieve the following IT security objectives and enforce the OSP; for that purpose, when IC physical security features are used, the specification of these physical security features shall be respected:

T.CLON*
The general threat is countered by the dedicated objective O.CLON*.

T.DIS_ES2
Illegal disclosure of ES is countered by O.DIS_MECHANISM2 and disclosure of Application by O.DIS_MEMORY*. More specifically this can be achieved by incorrect operation of the TOE, which is countered by O.OPERATE* and O.FLAW*, or by a direct observation during operation which is countered by O.SIDE

T.T_ES
Unauthorized use of TOE can be achieved by a degradation of the security mechanisms which is countered by O.OPERATE* and O.FLAW*, or by modification of the security mechanisms countered by O.TAMPER_ES . or by modification of TSF data, countered by O.MOD_MEMORY*.

T.T_CMD
To be able to have an unauthorized use of sequences sent to the TOE, it is necessary to achieve a degradation of the security mechanisms which is countered by O.OPERATE* and O.FLAW*, or to modify the security mechanisms countered by O.TAMPER_ES or by modification of TSF data, countered by O.MOD_MEMORY*.**T.MOD_SOFT***
The modification of embedded software of the TOE is countered by a correct operation of security mechanisms O.OPERATE* and O.FLAW*. The threat includes modification of the security mechanisms themselves which is countered by O.TAMPER_ES and modification of TSF data, countered by O.MOD_MEMORY*.

T.MOD_EXE
To be able to illegally execute programs on TOE, it is necessary to bypass or degrade the access security mechanisms. This is countered by O.OPERATE* and O.FLAW*. Modification of the security mechanisms is countered by O.TAMPER_ES. . It is also possible to gain access through modification of TSF data, which is countered by O.MOD_MEMORY*, or through disclosure of TSF data which is countered by O.DIS_MEMORY*.

T.MOD_LOAD
To be able to load illegally programs on TOE, it is necessary to achieve a degradation of the security mechanisms which is countered by O.OPERATE* and O.FLAW*, or to modify the security mechanisms countered by O.TAMPER_ES or by modification of TSF data, countered by O.MOD_MEMORY*.

T.MOD_SHARE
To be able to modify programs on TOE, it is necessary to bypass or degrade security mechanisms. This is countered by O.OPERATE* and O.FLAW*. Modification of the security mechanisms is countered by O.TAMPER_ES. Illegal Modification of Application data is countered by O.MOD_MEMORY*. This is also countered by protection of the confidentiality of TSF data: O.DIS_MEMORY*.

T.MOD_TSF
The illegal modification of TSF data of the TOE is countered by O.MOD_MEMORY* which addresses also TSF data. It is also possible to degrade or bypass access mechanisms, which is countered by O.TAMPER_ES and O.OPERATE*. Absence of design flaws, O.FLAW*, is necessary to counter the threat.

T.DIS_DATA
The disclosure of application user and TSF data on the TOE is countered by O.DIS_MEMORY*. To fulfill the threat, it is necessary to degrade or bypass access mechanisms, which is countered by O.SIDE and O.OPERATE*. Absence of design flaws, O.FLAW*, is necessary to counter the threat.

T.MOD_DATA
The modification of application user data on the TOE is countered by O.MOD_MEMORY*. To fulfill the threat, it can be necessary to degrade or bypass access mechanisms, which is countered by O.TAMPER_ES, O.OPERATE*. Absence of design flaws, O.FLAW*, is necessary to counter the threat.
.

Threats/OSP /Obj	ROLLBACK	RESOURCE	LOAD	SECURITY	EFFECT_L	REMOVE	EFFECT_R	SEGREGATE	CIPHER	DECIPHER
T.LOAD_MAN			X							
T.LOAD_APP			X							
T.LOAD_OTHER					X					
T.LOAD_MOD				X						
T.APP_DISC				X						
T.APP_CORR					X					
T.APP_REMOVE						X				
T.ERR_REMOVE						X				
T.DEL_REMOVE							X			
T.APP_READ								X		
T.APP_MOD								X		
T.RESOURCES	X	X								
OSP.CIPHER									X	
OSP.CONF-ALU										X

Table 7: Mapping of security objectives to threats & OSP relative to phases 6 and 7

The TOE shall use state of the art technology to achieve the following IT security objectives and enforce the OSP.

T.LOAD_MAN
O.LOAD imposes that application be loaded only with the permission of the administrator, which counters the threat.

T.LOAD_APP
O.LOAD controls the origin of the Loaded Application before loading, thus if necessary control is made by the administrator, it counters T.LOAD_APP.

T.LOAD_OTHER
Loading an application into an another illegal domain is countered by O.EFFECT_L, which prevents applications from having non-authorized effects on applications loaded in other domains.

T.LOAD_MOD
Alteration of Loaded Application during loading is prevented by O.SECURITY, which guarantees its integrity.

T.APP_DISC
Divulgence of Loaded Application during loading is prevented by O.SECURITY, which guarantees its confidentiality.

T.APP_CORR
Loading an application so it corrupts another application is countered by O.EFFECT_L, which prevents applications from having non-authorized effects on applications loaded in other domains.

T.APP_REMOVE
Removal of application without the consent of the administrator is countered by O.REMOVE, which imposes the authorization of the administrator.

T.ERR_REMOVE
Removal of application leaving confidential data is countered by O.REMOVE which imposes that the space left does not hold any information linked to remove application.

T.DEL_REMOVE
Deletion of part of a Loaded Application by removal of another is countered by O.EFFECT_R, which ensures that removal has no effect on other Loaded Applications.

T.APP_READ
Use of a Loaded Application to illegally read data contained in another application is countered by O.SEGREGATE which ensure that illegal reading of data of another application is not possible.

T.APP_MOD
Use of a Loaded Application to illegally modify data or code contained in another application is countered by O.SEGREGATE, which ensure that illegal modification of data or code of another application is not possible.

T.RESOURCES
Destruction or hoarding of card resources is prevented by O.ROLLBACK which guarantees that a failure does not compromise card resources and by O.RESOURCES which controls the use of card resources by Loaded Applications.

OSP.CIPHER
Cryptographic functions are required to actually protect the exchanged information (O.CIPHER). Remark that even if the TOE shall provide access to the appropriate TSFs, it is still the responsibility of the applications to use them.

OSP.CONF-ALU
KTU used to protect to encrypt sensitive sections of the ALU is decrypted by MULTOS (card) (O.DECIPHER)

5.2.2.2 Security objectives for the environment

The following descriptions and tables map the security objectives for the environment relative to the various threats countered, assumptions upheld and OSP enforced, in addition to the Smartcard PP.

ThreatsAssumptions /OSP/Obj	O.DEV_TOOLS*	O.DEV_DIS_ES	O.SOFT_DLV*	O.INIT_ACS	O.SAMPLE_ACS	O.DLV_ PROTECT	O.DLV_ AUDIT	O.DLV_RESP*	O.TEST_OPERATE*	O.USE_DIAG*	O.APPLI_DEV
T.CLON*		X	X	X	X						
T.DIS_INFO*		X									
T.DIS_DEL*			X								
T.DIS_ES1		X		X							
T.DIS_TEST_ES		X									
T.T_DEL*			X								
T.T_TOOLS	X										
T.T_SAMPLE2					X						
T.MOD_DEL*			X								
T.MOD		X		X							
A.DLV_PROTECT*						X					
A.DLV_AUDIT*							X				
A.DLV_RESP*								X			
A.USE_TEST*									X		
A.USE_PROD*									X		
A.USE_DIAG*										X	
A.APPLI_CONT											X

Table 8 : Mapping of security objectives for the environment to threats, assumptions and OSP relative to phase 1

T.CLON*
Cloning requires knowledge of:

- Development data and access to tools, which is countered by O.DEV_DIS_ES
- The software which is countered by O.SOFT_DLV*
- Initialization data which is countered by O.INIT_ACS

Cloning can also be done by using samples; this is countered by O.SAMPLES_ACS.

T.DIS_INFO*
Disclosure of IC assets is countered by O.DEV_DIS_ES, which guarantees the storage of classified information.

T.DIS_DEL*
Disclosure of embedded software and corresponding data during delivery is countered by O.SOFT_DLV*.

T.DIS_ES1
Disclosure of ES is countered by O.DEV_DIS_ES, which guarantees the storage of classified information, and by O.INIT_ACS which guarantees a controlled access to initialization data.

T.DIS_TEST_ES

Disclosure of ES test program is countered by O.DEV_DIS_ES, which guarantees the storage of classified information.

T.T_EL*
Theft of software delivered to IC manufacturer is countered by O.SOFT_DLV* which ensures trusted delivery.

T.T_TOOLS
Theft or unauthorized access to development tools is countered by O.DEV_TOOLS* which controls the accesses.

T.T_SAMPLE2
Theft of samples is countered by O.SAMPLE_ACS controlled access.

T.MOD_DEL*
Modification of software and related information is countered O.SOFT_DLV*.

T.MOD
Unauthorized modifications of software are countered by access control specified by O.DEV_DIS_ES and that of TSF data by O.INIT_ACS.

A.DLV_PROTECT*
Procedures shall ensure protection of TOE material/information under delivery and storage.

A.DLV_AUDIT*
Procedures shall ensure that corrective actions are taken in case of improper operation in the delivery process and storage.

A.DLV_RESP*
Procedures shall ensure that people dealing with the procedure for delivery have got the required skill.

A.USE_TEST*
It is assumed that appropriate functionality testing of the TOE is used in phases 4, 5 and 6.

A.USE_PROD*
It is assumed that security procedures are used during all manufacturing and test operations through phases 4, 5, 6 to maintain confidentiality and integrity of the TOE and of its manufacturing and test data (to prevent any possible copy, modification, retention, theft or unauthorized use).

A.USE_DIAG*
It is assumed that secure communication protocols and procedures are used between Smartcard and terminal.

A.APPLI_CONT
Whenever a Loaded-Application is to be loaded on the platform, it is assumed that its development and production follow the Administrator Guidance.

Threats	O.DLV_DATA	O.TEST_OPERATE*	O.DLV_PROTECT	O.DLV_AUDIT	O.DLV_RESP*	O.USE_DIAG*	O.APPLI_DEV
T.DIS_DEL1	X						
T.DIS_DEL2		X					
T.MOD_DEL1	X						
T.MOD_DEL2		X					
A.DLV_PROTECT*			X				
A.DLV_AUDIT*				X			
A.DLV_RESP*					X		
A.USE_TEST*		X					
A.USE_PROD*		X					
A.USE_DIAG*						X	
A.APPLI_CONT							X

Table 9 : Mapping of security objectives for the environment to threats, assumptions and OSP relative on delivery from phase 1 to phases 4 to 6

T.DIS_DEL1
Unauthorized disclosure of ES data during delivery is countered by O.DLV_DATA, which specifies a trusted delivery maintaining the confidentiality.

T.DIS_DEL2
Unauthorized disclosure of and ES data after delivery is countered by O.TEST_OPERATE* which specifies maintenance of the confidentiality.

T.MOD_DEL1
Unauthorized modification of ES data during delivery is countered by O.DLV_DATA, which specifies a trusted delivery maintaining the integrity.

T.MOD_DEL2
Unauthorized modification of ES data after delivery is countered by O.TEST_OPERATE* which specifies maintenance of the integrity and its test.

A.DLV_PROTECT*
Procedures shall ensure protection of TOE material/information under delivery and storage.

A.DLV_AUDIT*
Procedures shall ensure that corrective actions are taken in case of improper operation in the delivery process and storage.

A.DLV_RESP*
Procedures shall ensure that people dealing with the procedure for delivery have got the required skill.

A.USE_TEST*
It is assumed that appropriate functionality testing of the TOE is used in phases 4, 5 and 6.

A.USE_PROD*
It is assumed that security procedures are used during all manufacturing and test operations through phases 4, 5, 6 to maintain confidentiality and integrity of the TOE and of its manufacturing and test data (to prevent any possible copy, modification, retention, theft or unauthorized use).

A.USE_DIAG*
It is assumed that secure communication protocols and procedures are used between Smartcard and terminal.

A.APPLI_CONT
Whenever a Loaded-Application is to be loaded on the platform, it is assumed that its development and production follow the Administrator Guidance.

Threats/Assumptions/OSP	O.APPLI_DEV
T.LOAD_MOD	X
T.APP_DISC	X
T.DIS_DATA	X
T.CLON	X
A.APPLI_CONT	X

Table 10: Mapping of security objectives for the environment to threats, assumptions and OSPs on phases A1 and A2 (development and delivery for phase A1 to phases 6 and 7)

T.LOAD_MOD
Modification of Code and data of a Loaded-Application during its transfer and loading is countered by O.APPLI_DEV, which ensures the mechanisms to verify their integrity.

T.APP_DISC
Gaining access to confidential code and data of a Loaded-Application during its transfer and loading is countered by O.APPLI_DEV, which ensures the confidentiality.

T.DIS_DATA
Disclosure of sensitive User and TSF data is countered by is countered by O.APPLI_DEV, which requires that such data is stored encrypted.

T.CLON
Cloning may require the disclosure of sensitive User Data. Disclosure of such data is countered by i O.APPLI_DEV, which requires that such data is stored encrypted.

A.APPLI_CONT
Whenever a Loaded-Application is to be loaded on the platform, it is assumed that its development and production follow the Administrator Guidance.

5.2.3 Assumptions and Security Objectives for the Environment

This section demonstrates that the combination of the security objectives upholds or satisfies the identified assumptions for the operational environment.
Each of the assumptions for the environment is addressed by objectives.
Table 8 demonstrates which objectives contribute to the satisfaction of each assumption.
For clarity, the table does not identify indirect dependencies.

Phases		**Delivery process for phases 4 to 7**			**Phases 4 to 6**	**Phase 7**	**Phase A1**
	Assumptions /Objectives	**O.DLV_ PROTECT***	**O.DLV_ AUDIT***	**O.DLV_RESP ***	**O.TEST_OPE RATE***	**O.USE_DIAG ***	**O.APPLI_DE V**
4 to 7	A.DLV_PROTECT*	X					
4 to 7	A.DLV_AUDIT*		X				
4 to 7	A.DLV_RESP*			X			
4 to 7	A.USE_TEST*				X		
4 to 7	A.USE_PROD*				X		
7	A.USE_DIAG*					X	
A1	A.APPLI_CONT						X

Table 11: demonstrates mapping of security objectives for the operational environment to assumptions

A.DLV_PROTECT*
Procedures shall ensure protection of TOE material/information under delivery and storage. This assumption is upheld by **O.DLV_PROTECT*** in the delivery process for phases 4 to 7.

A.DLV_AUDIT*
Procedures shall ensure that corrective actions are taken in case of improper operation in the delivery process and storage. This assumption is upheld by **O.DLV_AUDIT*** in the delivery process for phases 4 to 7.

A.DLV_RESP*
Procedures shall ensure that people dealing with the procedure for delivery have got the required skill. This assumption is upheld by **O.DLV_RESP*** in the delivery process for phases 4 to 7.

A.USE_TEST*
It is assumed that appropriate functionality testing of the TOE is used in phases 4, 5 and 6. This assumption is upheld by **O.TEST_OPERATE***.

A.USE_PROD*
It is assumed that security procedures are used during all manufacturing and test operations through phases 4, 5, 6 to maintain confidentiality and integrity of the TOE and of its manufacturing and test data (to prevent any possible copy, modification, retention, theft or unauthorized use). This assumption is upheld by **O.TEST_OPERATE***.

A.USE_DIAG*
It is assumed that secure communication protocols and procedures are used between Smartcard and terminal. This assumption is upheld by **O.USE_DIAG*** in phase 7.

A.APPLI_CONT
Whenever a Loaded-Application is to be loaded on the platform, it is assumed that its development and production follow the Administrator Guidance. This assumption is upheld by **O.APPLI_DEV** in phase A1.

6. EXTENDED COMPONENTS DEFINITION

None.

7. SECURITY REQUIREMENTS

This chapter describes the security functional requirements for the TOE and the security assurance requirements for the TOE. To define the functional requirements, the supporting security infrastructure of the TOE is identified in the first section.

7.1 SUPPORTING SECURITY INFRASTRUCTURE

The following roles and responsibilities are assumed within the MULTOS security infrastructure

a) **MULTOS Security Manager (MSM):** defines and polices the MULTOS security infrastructure and provides criteria and services necessary for MULTOS participants to operate within the infrastructure. It acts as Certification Authority for the security infrastructure. It is assumed only one MSM exists. The MSM must be trusted by all participants in the infrastructure

b) **MULTOS Implementor:** the organization that implements a MULTOS version.
The MULTOS Implementor is licensed by MAOSCO and provides its MULTOS version to the IC Manufacturer. The MULTOS Implementor requests the MSM to provide MSM Controls Data, although this may be delivered to the MCD Manufacturer or MCD Issuer.

c) **Integrated Circuit (IC) Manufacturer:** manufacturer of silicon from which chips and smartcards are made. It is assumed the IC Manufacturer is trusted to perform its tasks correctly: This includes:

- To perform limited FLASH memory injection,
- The injection of diversified transport key (used by Initialization Mode). Security keys and data are provided by the MSM.

d) **Gemalto: is included in the new scheme in order to perform** the final manufacturing step by injecting the iKMA keys and data into the FLASH memory

The initialized ICs are provided to MCD Manufacturers:

e) **MULTOS Carrier Device (MCD) Manufacturer:** responsible for embedding the IC in its plastic carrier and for background printing on the card. The result is an initialized MCD. This operation is assumed not to be security sensitive. The MCD Manufacturer may also receive MSM Controls Data from the MSM and enable the MCDs. Initialized and enabled MCDs are provided to MCD Issuers.

c) **MCD Issuer:** responsible for issuing to users the MCD itself. The MCD Issuer may also enable initialized MCDs, by loading MSM Controls Data received from the MSM onto the MCDs. MCD Issuers retain the ultimate authority over what applications are loaded on their MCDs. MCD Issuers register applications with the MSM, provide information related to the applications and receive application load and delete certificates from the MSM.

d) **Application Writer:** licensed by MAOSCO to produce applications for MULTOS. Supplies applications under contract to Application Issuers.

e) **Application Issuer:** an organization that wishes to offer an application to MCD Users.
The Application Issuer agrees with an MCD Issuer that the application can be loaded onto MCDs belonging to the MCD Issuer.

f) **Application Provider:** the organization that takes responsibility for an application, by certifying it with the organization's public key and encrypting it where necessary. The Application Provider is a role that can be performed by an Application Writer, Application Issuer or MCD Issuer, rather than necessarily, being an organization in its own right.

g) **Application Loader:** responsible for performing the technical operation of loading applications onto MCDs. The Application Loader enters into an agreement with one or more Application Issuers and MCD Issuers for loading applications supplied by one or more Application Providers.

h) **MCD User:** final user of the MCD.

The MSM authorizes potential MULTOS platforms (known as MA-cards). To receive MSM authorization, a platform must comply with criteria covering attributes of the platform itself and the procedures associated with its manufacture.

7.2 SECURITY FUNCTIONAL REQUIREMENTS (SFRS)

This section defines the functional requirements for the TOE using only functional requirement components drawn from the CC part 2.
The assignment and selection operations are written in **bold style** for a better readability.

7.2.1 Security Audit Automatic Response (FAU_ARP)

7.2.1.1 FAU_ARP.1 Security alarms

FAU_ARP.1.1 Mute Iteration. The TSF shall take action to cause **the MCD to mute** upon detection of a potential security violation.

FAU_ARP.1.1 Shutdown Iteration. The TSF shall take action to cause **the MCD to enter Shutdown mode** upon detection of a potential security violation.

7.2.2 Security audit analysis (FAU_SAA)

7.2.2.1 Potential violation analysis

FAU_SAA.1.1 Mute Iteration. The TSF shall be able to apply a set of rules in monitoring the audited events and based upon these rules indicate a potential violation of the enforcement of the SFRs.

FAU_SAA.1.2 Mute Iteration. The TSF shall enforce the following rules for monitoring audited events:

a) Accumulation or combination of:

- **Hardware fault – one of:**
-
- **Sensor test failure during processing of first command after reset or subsequent commands.**
- **NVM write failure.**
- **SLC52 MMU failure.**
- **Application code integrity failure when application is selected or is run for the first time following a reset.**
- **Integrity check failure over the MSM Controls Data or security data held within the FLASH memory of MULTOS**

b) none. Known to indicate a potential security violation.

b) **none**.

FAU_SAA.1.1 Shutdown Iteration. The TSF shall be able to apply a set of rules in monitoring the audited events and based upon these rules indicate a potential violation of the TSP.

FAU_SAA.1.2 Shutdown Iteration. The TSF shall enforce the following rules for monitoring audited events:

a) Accumulation or combination of:

- **MSM retry counter has expired**
- **Corrupted fault detection counters**
- **Corrupted retry counters**
- **Flash loader failure**
- **General fault (e.g. assert failure)**

known to indicate a potential security violation.

b) **none**.

7.2.3 Cryptographic key management (FCS_CKM)

7.2.3.1 FCS_CKM.1 Cryptographic key generation

FCS_CKM.1.1 The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **[assignment: cryptographic key generation algorithm]** and specified cryptographic key sizes **[assignment: cryptographic key sizes]** that meet the following: **[assignment: list of standards]**.

Iteration	Algorithm	Key size	Standards
/RSA CRT	RSA CRT key generation	1024, 1536, 2048, 4096	ANSI X9.31
/ECFP	ECC key generation	160, 192, 224, 256, 320, 384, 512, 521	ANSI X9.62

7.2.3.2 FCS_CKM.3 Cryptographic key access

FCS_CKM.3.1. The TSF shall perform **a read of cryptographic key** in accordance with a specified cryptographic key access method, **a temporary copy key in RAM** that meets the following: **none**.

7.2.3.3 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 **physical irreversible destruction of the stored key value** that meets the following: **none**.
.

7.2.4 FCS_COP Cryptographic operations

7.2.4.1 FCS_COP.1 Cryptographic operations

The TSF shall perform **[assignment: list of cryptographic operations]** in accordance with a specified cryptographic algorithm **[assignment: cryptographic algorithm]** and cryptographic key sizes **[assignment: cryptographic key sizes]** that meet the following: **[assignment: list of standards]**.

Iteration	operation	algorithm	Key size	Standards
/RSA-SIGN	signature & verification	RSA (STD) RSA CRT	1024, 1152, 1280, 1536 and 2048 3072, 4096	[ISO9796-2] RSA SHA PKCS#1 RSA SHA PKCS#1 PSS
/RSA-CIPHER	Encryption & decryption	RSA (STD) RSA CRT	1024, 1152, 1280, 1536 and 2048 3072, 4096	[ISO9796-2] RSA SHA PKCS#1
/ECC-SIGN	signature & verification	ECC	160, 192, 224, 256, 320, 384, 512, 521	[TR-03111] ECDSA SHA
/TDES-CIPHER	Encryption & decryption	TDES	112 168	[SP800-67] [ISO9797-1] DES NOPAD DES PKCS#5 DES 9797 M1 M2
/AES-CIPHER	Encryption & decryption	AES	128, 192, 256	[FIPS197] AES 128 NOPAD
/TDES-MAC	Signature, Verification	TDES	112 168	[SP800-67] [ISO9797-1] DES MAC ISO9797-1 M1 M2 Alog3 DES MAC NOPAD DES MAC PKCS#5
/SHA	Hashing	Hashing	SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	NA

7.2.5 Access control policy FDP_ACC

7.2.5.1 FDP_ACC.2 Complete access control

FDP_ACC.2.1 Load Application SFP Iteration. The TSF shall enforce the **Load Application SFP** on **MULTOS ES and Application Load Certificate**, and all operations among subjects and objects covered by the SFP.

FDP_ACC.2.1 Delete Application SFP Iteration. The TSF shall enforce the **Delete Application SFP** on **MULTOS** ES and Application Delete Certificate, and all operations among subjects and objects covered by the SFP.

FDP_ACC.2.2. 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.

7.2.6 Access control functions FDP_ACF

7.2.6.1 FDP_ACF.1 Security attribute based access control

FDP_ACF.1.1 Load Application SFP Iteration. The TSF shall enforce the **Load Application SFP** to objects based on **Unique Application Identifier present in the ALC, Unique Application Identifier of loaded-applications, MCD Enabled Flag, Application Load Permissions, MCD Load Permissions, and History List.**

FDP_ACF.1.2 Load Application SFP Iteration. The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed. **See the table below.**

FDP_ACF.1.3 Load Application SFP Iteration. The TSF shall explicitly authorize access of subjects to objects based on the following additional rules. **See the table below.**

FDP_ACF.1.4 Load Application SFP Iteration. The TSF shall explicitly deny access of subjects to objects based on the **table below.**

Security attributes	Governing access rules	Authorizing access rules	Denying access rules
Unique Application Identifier present in the ALC and Unique Application Identifier of loaded-applications	Verify there is other application currently loaded on this MCD with the same Application Identifier.	Establish that there is no other application currently loaded on this MCD with the same Application Identifier. Application load process continues.	Establish that there is another application currently loaded on this MCD with the same Application Identifier. Application load process is aborted.
MCD enabled flag	Verify the MCD is enabled (ie that the MSM Controls Data for this MCD has been installed)	Establish that the MCD is enabled (ie that the MSM Controls Data for this MCD has been installed). Application load process continues.	Establish that the MCD is not enabled (ie that the MSM Controls Data for this MCD has not been installed). Application load process is aborted.
Application Load Permissions and MCD Load Permissions	Verify the application load permissions are compatible with the MCD permissions which were installed when the card was enabled	Establish that the application load permissions are compatible with the MCD permissions which were installed when the card was enabled. Application load process continues.	Establish that the application load permissions are not compatible with the MCD permissions which were installed when the card was enabled. Application load process is aborted.
History list	Determine if the application is being re-load a second time on to this MCD, and whether that is permitted	If the application is being re-load a second time on to this MCD, and that is permitted. Application load process continues.	If the application is being re-load a second time on to this MCD, and that is not permitted. Application load process is aborted.

FDP_ACF.1.1 Delete Application SFP Iteration. The TSF shall **enforce the Delete application SFP** to objects based on **Unique Application Identifier present in the ADC, Unique Application Identifier of loaded-applications, Application Load Permissions, MCD Load Permissions.**

FDP_ACF.1.2 Delete Application SFP Iteration. The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed. **See the table below.**

FDP_ACF.1.3 Delete Application SFP Iteration. The TSF shall explicitly authorize access of subjects to objects based on the following additional rules. **See the table below.**

FDP_ACF.1.4 Delete Application SFP Iteration. The TSF shall explicitly deny access of subjects to objects based on the **table below.**

Security attributes	Governing access rules	Authorizing access rules	Denying access rules
Unique Application Identifier present in the ADC and Unique Application Identifier of loaded-applications	Verify an application with the Application Identifier specified in the ADC is loaded on this MCD	Establish that an application with the Application Identifier specified in the ADC is loaded on this MCD. Application deletion process continues.	Establish that no application with the Application Identifier specified in the ADC is loaded on this MCD. Application deletion process is aborted.
Application Load Permissions and MCD Load Permissions	Verify the application permissions are compatible with the MCD permissions which were installed when the card was enabled	Establish that the application permissions are compatible with the MCD permissions, which were installed when the card was enabled. Application deletion process continues.	Establish that the application permissions are not compatible with the MCD permissions, which were installed when the card was enabled. Application deletion process is aborted.

7.2.7 Data authentication FDP_DAU

7.2.7.1 FDP_DAU.1 Basic data authentication

FDP_DAU.1.1. The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of **application's code spaces.**

FDP_DAU.1.2. The TSF shall provide **MULTOS** with the ability to verify evidence of the validity of the indicated information.

7.2.8 Import from outside TSF control FDP_ITC

7.2.8.1 FDP_ITC.1 Import of user data without security attributes

FDP_ITC.1.1. The TSF shall enforce the **Load Application SFP** when importing user data, controlled under the SFP, from outside of the TSC.

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

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

7.2.9 Residual information protection FDP_RIP

7.2.9.1 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 **de-allocation of the resource from** the following objects: **application's code and data spaces.**

7.2.10 Rollback (FDP_ROL)

7.2.10.1 FDP_ROL.1 Basic rollback

FDP_ROL.1.1. The TSF shall enforce **Load Application SFP** to permit the rollback of the **load of an application** on the **application's code and data.**
FDP_ROL.1.2. The TSF shall permit operations to be rolled back within **a failure occurs during loading of an application.**

7.2.11 Stored data integrity (FDP_SDI)

7.2.11.1 FDP_SDI.2 Stored data integrity monitoring and action

FDP_SDI.2.1. The TSF shall monitor user data stored in container controlled by TSF for **memory corruption** on all objects, based on the following attributes: **four-byte check sum**.

FDP_SDI.2.2. Upon detection of a data integrity error, the TSF shall **abend the current session**.

7.2.12 Authentication failures (FIA_AFL)

7.2.12.1 FIA_AFL.1 Authentication failure handling

FIA_AFL.1.1. The TSF shall detect when **20** unsuccessful authentication attempts occur related to **execution of SetMSMControls command, DeleteMELApplication command and CreateMELApplication command.**

FIA_AFL.1.2. When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall **permanently disable the incriminated command.**

7.2.13 User attribute definition (FIA_ATD)

7.2.13.1 FIA_ATD.1 User attribute definition

FIA_ATD.1.1. The TSF shall maintain the following list of security attributes belonging to individual users: **See the table below**.

MULTOS Security Manager	MCD Enabled Flag
	History List Entry
MCD Issuer	MCD Issuer Identifier
	Unique Application Identifier
	Application Load Permission
	MCD Permissions

7.2.14 User Authentication (FIA_UAU)

7.2.14.1 FIA_UAU.1 Timing of authentication

FIA_UAU.1.1. The TSF shall allow **processing of Check Data command** on behalf of the user to be performed before the user is authenticated.

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.

7.2.14.2 FIA_UAU.4 Single-use Authentication Mechanisms

FIA_UAU.4.1. The TSF shall prevent reuse of authentication data related to **application's load and delete authentication mechanisms.**

7.2.15 User identification (FIA_UID)

7.2.15.1 FIA_UID.1 Timing of identification

FIA_UID.1.1. The TSF shall allow **processing of Check Data command** on behalf of the user to be performed before the user is identified.

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.

7.2.16 User-subject Binding (FIA_USB)

7.2.16.1 FIA_USB.1 User-subject binding

MULTOS Security Manager	MCD Enabled Flag
	History List Entry
MCD Issuer	MCD Issuer Identifier
	Unique Application Identifier
	Application Load Permission
	MCD Permissions

FIA_USB.1.1 The TSF shall associate the following user security attributes with subjects acting on the behalf of that user: See table in section 7.2.6.1 **(FDP_ACF.1)**

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: See table in section 7.2.13 (**FIA_ATD.1.1)**

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:
No changes are permitted.

7.2.17 Management of function in the TSF (FMT_MOF)

7.2.17.1 FMT_MOF.1 Management of security functions behavior

FMT_MOF.1.1. The TSF shall restrict the ability to **determine the behavior of** the functions **Application Load Certificate Control SF and Application Deletion Certificate Control SF** to **MSM**.

FMT_MOF.1.1. The TSF shall restrict the ability to **enable** the functions **Application Load Certificate Control SF and Application Deletion Certificate Control SF** to **MSM**.

7.2.18 Management of security attributes (FMT_MSA)

7.2.18.1 FMT_MSA.1 Management of security attributes

FMT_MSA.1.1. The TSF shall enforce the **Load Application SFP and Delete Application SFP** to restrict the ability to **load** the following security attributes to **the MSM**:

- **MCD Issuer Product Identifier.**
- **MCD Issuer Identifier.**
- **MCD Batch Number.**
- **RFU 2 (Reserved for Future Use).**
- **RFU 4.**
- **RFU 5.**
- **RFU 6.**
- **MCD-unique Identifier.**
- **Asymmetric transport key set (MKD).**

7.2.18.2 FMT_MSA.2 Secure security attributes

FMT_MSA.2.1. The TSF shall ensure that only secure values are accepted for security attributes.

7.2.18.3 FMT_MSA.3 Static attribute initialization

FMT_MSA.3.1. The TSF shall enforce the **Load Application SFP and Delete Application SFP** to provide **MSM Controls Data** default values for security attributes that are used to enforce the SFP.

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

7.2.19 Management of TSF data (FMT_MTD)

7.2.19.1 FMT_MTD.1 Management of TSF data

FMT_MTD.1.1. The TSF shall restrict the ability to **load** the **MSM Controls Data** to **MSM**.

7.2.19.2 FMT_MTD.2 Management of limits on TSF data

FMT_MTD.2.1. The TSF shall restrict the specification of the limits for **MSM Controls Data** to **MSM**.

FMT_MTD.2.2. The TSF shall take the following actions, if the TSF data are at, or exceed, the indicated limits: **MCD becomes mute**.

7.2.20 Security management roles (FMT_SMR)

7.2.20.1 FMT_SMR.1 Security roles

FMT_SMR.1.1. The TSF shall maintain the roles:
MULTOS Security Manager (MSM)
MCD Issuer
Application Provider

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

7.2.21 Unobservability (FPR_UNO)

7.2.21.1 FPR_UNO.1 Unobservability

FPR_UNO.1.1. The TSF shall ensure that **any users** are unable to observe the **cryptographic** operations on **Application Load Certificate, Application Delete Certificate, Application Load Unit, MSM Controls Data and hash digest of the contents of a selected area of MCD's memory** by **MULTOS**.
The functional requirement must be understood in the sense of protection against observation of the mechanisms and TSF data used and of User data manipulated during the operation. The intent is to protect against side channel attacks.

7.2.22 Fail secure (FPT_FLS)

7.2.22.1 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:

a) An apparent corruption of the MSM Controls Data or security data held within the FLASH memory of MULTOS

b) An unexpected hardware event occurred

c) MULTOS determines that it has executed an invalid sequence of instructions (possibly due to electromagnetic or mechanical interference)

d) A critical process is interrupted

e) There have been too many failed attempts to load MSM Controls Data.

7.2.23 TSF Physical protection (FPT_PHP)

7.2.23.1 FPT_PHP.3 Resistance to physical attack

FPT_PHP.3.1. The TSF shall resist the **following physical tampering scenarios** to the **following list of TSF devices/elements** by responding automatically such that the TSP is not violated.

Physical tampering scenarios	TSP devices/elements
Abnormal use of reset signal	All TSF devices/elements
Abnormal use of power signal	All TSF devices/elements
Clock rate variations	The processor
Dynamic power analysis	Cryptographic operations

7.2.24 Trusted recovery (FPT_RCV)

7.2.24.1 FPT_RCV.4 Function recovery

FPT_RCV.4.1. The TSF shall ensure that the **following list of functions and failure scenarios** have the property that the SF either completes successfully, or for the indicated failure scenarios, recovers to a consistent and secure state.

Security functions	Failure scenarios
Application Load Certificate Control SF	Reset/power down during command processing
Application Delete Certificate Control SF	Reset/power down during command processing Too many failed Delete Command
Unprotected/Protected Application Load Unit SF	Reset/power down during command processing Too many failed Create Command
Confidential Application Load Unit SF	Reset/power down during command processing Too many failed Create Command
MSM Controls Data Load Management SF	Reset/power down during command processing Too many failed Set MSM Controls Command
Critical Data Overwrite SF	Reset/power down during command processing or application execution
Reset Protection SF	Reset/power down during command processing or application execution
Integrity Checks SF	Reset/power down during command processing or application execution
Start-up Validity Checks and Initialization SF	Reset/power down during command processing or application execution
All Security Functions	FLASH memory write failure Power loss Integrity failure

7.2.25 Inter-TSF TSF data consistency (FPT_TDC)

7.2.25.1 FPT_TDC.1 Inter-TSF basic TSF data consistency

FPT_TDC.1.1. The TSF shall provide the capability to consistently interpret **Application Load Certificate, Application Delete Certificate, Key Transformation Unit, Application Provider Signature and MSM Controls Data** when shared between the TSF and another trusted IT product.

FPT_TDC.1.2. The TSF shall use **signatures format on the certificates, the Application Load Unit, the Key Transformation Unit and MSM Controls Data** when interpreting the TSF data from another trusted IT product.

7.2.26 TSF self-test (FPT_TST)

7.2.26.1 FPT_TST.1 TSF Testing

FPT_TST.1.1. The TSF shall run a suite of self-tests **at the conditions when MULTOS is powered-up or reset** to demonstrate the correct operation of some parts of the TSF. These self-tests include checks that FLASH memory is writable, that the chip's active shield is operational and that the chip-level sensor self-tests pass.

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

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

7.2.27 Resource allocation (FRU_RSA)

7.2.27.1 FRU_RSA.1 Maximum quotas

FRU_RSA.1.1. The TSF shall enforce maximum quotas of the following resources: **FLASH memory and RAM** that **applications, functions, codelets and primitives** can use **simultaneously**.

7.3 SECURITY ASSURANCE REQUIREMENTS (SARS)

This section describes the SARs. The Assurance requirement is EAL5 augmented with additional assurance components listed in the following section. These components are hierarchical ones to the components specified in EAL5.

7.3.1 ALC_DVS.2: Sufficiency of Security Measures

Developer action elements:
ALC_DVS.2.1D. The developer shall produce development security documentation.
Content and presentation of evidence elements:

ALC_DVS.2.1C. The development security documentation shall describe all the physical, procedural, personnel, and other security measures that are necessary to protect the confidentiality and integrity of the TOE design and implementation in its development environment.

ALC_DVS.2.2C. The development security documentation shall provide evidence that these security measures are followed during the development and maintenance of the TOE.

ALC_DVS.2.3C. The evidence shall justify that the security measures provide the necessary level of protection to maintain the confidentiality and integrity of the TOE.
Evaluator action elements:

ALC_DVS.2.1E. The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_DVS.2.2E. The evaluator shall confirm that the security measures are being applied.
Dependencies:

No dependencies.

7.3.2 AVA_VAN.5: Advanced Methodical Vulnerability Analysis

Developer action elements:
AVA_VAN.5.1D The developer shall provide the TOE for testing.
Content and presentation elements:
AVA_VAN.5.1C The TOE shall be suitable for testing.
Evaluator action elements:
AVA_VAN.5.1E The evaluator ***shall confirm*** that the information provided meets all requirements for content and presentation of evidence.
AVA_VAN.5.2E The evaluator ***shall perform*** a search of public domain sources to identify potential vulnerabilities in the TOE.
AVA_VAN.5.3E The evaluator ***shall perform*** an independent, methodical vulnerability analysis of the TOE using the guidance documentation, functional specification, TOE design, security architecture description and implementation representation to identify potential vulnerabilities in the TOE.
AVA_VAN.5.4E The evaluator ***shall conduct*** penetration testing based on the identified potential vulnerabilities to determine that the TOE is resistant to attacks performed by an attacker possessing **High** attack potential.
Dependencies:
ADV_ARC.1 Security architecture description.
ADV_FSP.2 Security-enforcing functional specification.
ADV_TDS.3 Basic modular design.
ADV_IMP.1 Implementation representation of the TSF.
AGD_OPE.1 Operational user guidance.
AGD_PRE.1 Preparative procedures.

7.4 SECURITY FUNCTIONAL REQUIREMENTS DEPENDENCIES

This section demonstrates that all dependencies between components of security functional requirements included in this PP are satisfied.
Table 9 lists all functional components including security requirements in the IT environment. For each component, the dependencies specified in Common Criteria are listed, and a reference to the component number is given.

Number	Security functional requirements	Dependencies	Line N°
1	FAU_SAA.1: Potential Violation Analysis	FAU_GEN.1	*
2	FCS_CKM.1: Cryptographic Key Generation	FDP_ITC.1, FCS_CKM.4, FMT_MSA.2	9, 4, 20
3	FCS_CKM.3: Cryptographic Key Access	FDP_ITC.1, FCS_CKM.4, FMT_MSA.2	9, 4, 20
4	FCS_CKM.4: Cryptographic Key Destruction	FDP_ITC.1 , FMT_MSA.2	9, 20
5	FCS_COP.1: Cryptographic Operation	FDP_ITC.1, FCS_CKM.4, FMT_MSA.2	9, 3, 20
6	FDP_ACC.2: Complete Access Control	FDP_ACF.1	7
7	FDP_ACF.1: security attributes based Access Control	FDP_ACC.1, FMT_MSA.3	H(5), 21
8	FDP_DAU.1: basic Data Authentication	none	
9	FDP_ITC.1: Import of user data without security attributes	FDP_ACC.1,FMT_MSA.3	H(5), 21
10	FDP_RIP.1: subset residual information protection	none	
11	FDP_SDI.2: stored data integrity monitoring and action	none	
12	FIA_AFL.1: Authentication failure handling	FIA_UAU.1	14
13	FIA_ATD.1: User attribute definition	None	
14	FIA_UAU.1: Timing of authentication	FIA_UID.1	16
15	FIA_UAU.4: Single-use authentication mechanisms	none	
16	FIA_UID.1: timing of identification	none	
17	FIA_USB.1: user-subject binding	FIA_ATD.1	13
18	FMT_MOF.1: management of security functions behavior	FMT_SMR.1	23
19	FMT_MSA.1: management of security attributes	FDP_ACC.1, FMT_SMR.1	H(5), 23
20	FMT_MSA.2: Secure security attributes	ADV_SPM.1, FSP_ACC.1, FMT_MSA.1, FMT_SMR.1	by EAL5 H(5), 19, 23
21	FMT_MSA.3: Secure attributes initialization	FMT_MSA.1, FMT_SMR.1	19, 23
22	FMT_MTD.1: management of TSF data	FMT_SMR.1	23
23	FMT_SMR.1: security roles	FIA_UID.1	16
24	FPR_UNO.1: Unobservability	none	

Number	Security functional requirements	Dependencies	Line N°
25	FPT_FLS.1: failure with preservation of secure state	ADV_SPM.1	by EAL5
26	FPT_PHP.3: Resistance to physical attack	none	
28	FPT_TDC.1: inter-TSF basic TSF data consistency	none	
29	FPT_TST.1: TSF testing	none	
30	FAU_ARP.1: Security Alarms	FAU_SAA.1	1
31	FDP_ROL.1: Basic Rollback	FDP_ACC.1	H(5)
32	FMT_MTD.2: Management of limits on TSF data	FMT_MTD.1, FMT_SMR.1	22, 23
33	(not used)		
34	(not used)		
35	FRU_RSA.1: Maximum quotas	none	

Table 12: Functional dependencies in Multi-Application environment

*: Dependencies are not met for the reasons given below.
H(5) means that the dependency is satisfied by a higher hierarchical component.
Table 9 shows that the functional component dependencies are satisfied by all functional components of the PP except for the components stated in bold characters, as explained as follows:
The dependency of FAU_SAA.1 with FAU_GEN.1 is not applicable to the TOE; the FAU_GEN.1 component forces many security relevant events to be recorded (due to dependencies with other functional security components) and this is not achievable in a smartcard since many of these events result in card being in an insecure state where recording of the event itself could cause a security breach. It is then assumed that the function FAU_SAA.1 may still be used and the specific audited events will have to be defined in the ST independently with FAU_GEN.1.

7.5 SECURITY REQUIREMENTS RATIONALE

7.5.1 Security Functional Requirements Rationale

This section demonstrates that the combination of the security requirements (TOE and environment) is suitable to satisfy the identified security objectives and that it can be traced back to the security objectives.

7.5.1.1 SFRs Tracing Rationale

In this sub-section Table 10 traces and demonstrates which security functional requirement contributes to the satisfaction of each TOE security objective. For clarity, the table does not identify indirect dependencies.

Security Functional Requirements	O.TAMPER_ES	O.SIDE	O.OPERATE*	O.DIS_MECHANISM2	O.DIS.MEMORY*	O.MOD_MEMORY*	O.FLAW*	O.CLON*
EAL5Requirements							X	
FAU_ARP.1	X		P	P	X	X		
FAU_SAA.1	X		P	P	X	X		
FCS_CKM.1	X		P		P	P		P
FCS_CKM.3	X		P		P	P		P
FCS_CKM.4	X		P		P	P		X
FCS_COP.1	X				X	X		P
FDP_ACC.2	X		P	X	X	X		P
FDP_ACF.1	X		P	X	X	X		P
FDP_DAU.1	X		P		X	X		P
FDP_ITC.1					X	X		
FDP_RIP.1	X				P			
FDP_SDI.2			P			X		
FIA_AFL.1	X		P					P
FIA_ATD.1	X		P					
FIA_UAU.1	X				X	X		P
FIA_UAU.4	X				X	X		P
FIA_UID.1	X				X	X		P
FIA_USB.1	X				X	X		P
FMT_MOF.1	X		X	X	P	P		P
FMT_MSA.1	X		P	X	P	P		P
FMT_ MSA.2	X		P	X	P	P		P
FMT_ MSA.3	X		P	X	P	P		P
FMT_MTD.1					X	X		P
FMT_SMR.1	X		X					
FPR_UNO.1		X	P		X			X
FPT_FLS.1	X							
FPT_PHP.3	X		X	X	X	X		X
FPT_TDC.1	X					X		
FPT_TST.1			X		X	X		

Security Functional Requirements	ROLLBACK	RESOURCE	LOAD	SECURITY	EFFECT_L	REMOVE	EFFECT_R	SEGREGATE	CIPHER	DECIPHER
FAU_ARP.1		X								
FAU_SAA.1		X								
FCS_CKM.1				X		X			X	
FCS_CKM.3				X		X			X	
FCS_CKM.4				X		X			X	
FCS_COP.1			X	X		X			X	
FCS_COP.1/RSA										X
FDP_ACC.2			X			X		X		
FDP_ACF.1					X		X	X		
FDP_ITC.1			X	X						
FDP_RIP.1						X				
FDP.ROL.1	X									
FIA_UID.1			X			X				
FIA_UAU.1			X			X				
FIA_UAU.4										
FMT_MSA.1			X			X				
FMT_ MSA.2			X			X				
FMT_ MSA.3			X			X				
FMT_MTD.1										
FMT_MTD.2		X	X							
FMT_SMR.1			X			X				
FPT_FLS.1	X				X		X	X		
FPT.RCV.4	X				X		X			
FRU_RSA.1		X								

Legend: P: Partial ; X: relevant

Table 13: Mapping of security functional requirements and objectives

7.5.1.2 SFRs Justifications Rationale

This sub-section describes and justifies how the security objectives for the TOE are met by the security functional requirements.
The assurance requirements contribute to the satisfaction of the O.FLAW* security objective. They are suitable because they provide the assurance that the TOE is designed, implemented and operates so that the IT security requirements are correctly provided.

O.TAMPER_ES
This objective is met through:

Protection of critical parts from tampering:
If a failure occurs, a secure state is preserved (FPT_FLS.1), so critical parts are preserved. Resistance to physical attack (FPT_PHP.3) also allows protection of critical parts. Inter-TSF basic TSF data consistency (FPT_TDC.1) also allows better protection from tampering.

Prevention of unauthorized changes:
Identification and authentication: Thanks to FIA_ATD.1 and FIA_USB.1 only authorized users are able to load certificates, ALU or MSM Controls Data. The only command available before Enablement is the CheckData command which permits only the verification of the validity of an MCD, it does not allow any unauthorized changes.
Since applications can only be loaded after enablement, the timing of authentication by presentation of the ALC restricted (FIA_UAU.1).
The realization of FIA_UAU.4 in the form of the history list feature of SF1 means that the authentication of ALCs and ADCs can be further strengthened to allow them to be used only once. Moreover, brute force attacks are prevented with FIA_AFL.1. Cryptographic aspects do not permit any unauthorized changes.
Security management: Only two security roles are allowed (FMT_SMR.1) which decreases the risk of unauthorized changes of the TOE. MULTOS Security Manager is the only authorized role able to manage security functions behavior via the security attributes contained in the MSM Controls Data (FMT_MOF.1, FMT_MSA.1, FMT_MSA.3). Application Provider is the only authorized role able to sign application certificate.

Protection of parameters and keys:
Cryptographic keys are temporarily copied in RAM (FCS_CKM.1, FCS_CKM.3) to be used for cryptographic operations (FCS_CKM.4) and then they are destroyed (FCS_COP.1) to prevent an unauthorized user gaining access to them.
Thanks to FDP.ACC.2 and FDP_ACF.1, only authorized operations are allowed between subjects and objects defined in the Access Control Policy. It prevents fake Load and Delete Certificate being loaded onto an MCD. Moreover, FDP_DAU.1 and FDP_RIP.1 do not allow the application's code and data to be modified or disclosed.
If a potential violation is detected, the MCD will abend current operation and become mute or it will shutdown to prevent critical parts being disclosed. FAU_SAA.1 and FAU_ARP.1 define potential violations and actions to be taken in each case to ensure protection of critical parts.

O.SIDE
Interpretation of side channel information leakage is countered by FPR_UNO, which ensures that observation of signals cannot reveal information that could allow illegal access and operations.

O.OPERATE*
Correct operation of security functions is assured by:
Security management: Only security roles defined in FMT_SMR.1 are able to provide security management. It is by virtue of FMT_MOF.1 that the MSM retains management of security functions through the control the MSM has over generation of MSM Controls Data and ALCs and ADCs.
MSM ensures management of security functions behavior. In this way, operations of security functions are secured to be determined by authorized users.
Protection of TSF: Resistance to physical attacks (FPT_PHP.3) and TSF testing (FPT_TST) permit to protect security data which security functions are reliant on.
In this way, correct operations of security functions are ensured.
On a second level, other SFR are active: FAU_ARP.1, FAU_SAA.1, FCS_CKM.1, FCS_CKM.3, FCS_CKM.4, FDP_ACC.2, FDP_ACF.1, FDP_DAU.1, FDP_SDI.2, FIA_AFL.1, FIA_ATD.1, FMT_MSA.1, FMT_MSA.2, FMT_MSA.3, and FPR_UNO.1.

O.DIS_MECHAN2
Protection of security mechanisms against unauthorized disclosure is assured by the following:
Protection of TSF: Resistance to physical attack (FPT_PHP.3) prevent unauthorized users to directly or indirectly observe the security mechanisms by using physical attack.
Security management: Only MULTOS Security Manager is able to determine the behavior and enable (FMT_MOF.1, FMT_MSA.3) security mechanisms by loading security attributes contained in MSM Controls Data (FMT_MSA.1). MSM Controls Data is encrypted by MSM to ensure only secure values are accepted for security attributes (FMT_MSA.2).
User data protection: FDP_ACC.2 and FDP_ACF.1 ensure user data is not corrupted (in order to permit security mechanisms unauthorized disclosure). In the same way, FAU_SAA.1 and FAU_ARP.1 allow tracking of possible attacks against user data.

O.DIS_MEMORY*
User data protection: FDP_ACC.2 and FDP_ACF.1 are used to ensure that applications are not loaded on a fake MCD. FDP_DAU.1 guarantees the validity of an application's code space and to ensure MULTOS will not send responses that may be based on corrupted data. FDP_ITC.1 and partially FDP_RIP.1 fulfil this objective.
Authentication and identification: FIA_UAU.1, FIA_UID.1 and FIA_USB.1 allow only authorized users to access memory. FIA_UAU.4 guarantees the authenticity of the application thanks to single-use authentication mechanisms.
Cryptographic support: FCS_COP.1, FCS_CKM.1, FCS_CKM.3 and FCS_CKM.4 are used for authentication.
Security management: Only MSM is authorized to load encrypted MSM Controls Data (FMT_MTD.1). Because of the encryption, an unauthorized user cannot disclose this data. FMT_MOF.1, FMT_MSA.1, FMT_MSA.2 and FMT_MSA.3 are used as support.
Unobservability: FPR_UNO.1 is used so that data is not revealed during operations.
Protection of the TSF: FPT_PHP.3 prevents unauthorized users disclosing sensitive information in memories using physical attack. FPT_TST.1 provides initializations of memories, when the MCD is powered-up or reset, to prevent unauthorized disclosure of data.
Security audit: FAU_ARP.1 and FAU_SAA.1 monitor problems related to unauthorized disclosure of sensitive information.

O.MOD_MEMORY*
User data protection: FDP_DAU.1 and FDP_SDI.2 ensure that the application's code has not been corrupted or modified by an unauthorized user. FDP_ITC.1 and partially FDP_ACC.2, FPT_TDC.1 and FDP_ACF.1 also fulfil this objective.
Authentication and identification: FIA_UAU.1, FIA_UID.1 and FIA_USB.1 allow only authorized users to access memory. FIA_UAU.4 guarantees the authenticity of the application thanks to single-use authentication mechanisms. This objective is supported partially by FIA_AFL.1 and FIA_ATD.1 and FPT_TDC.1.
Cryptographic support: FCS_COP.1, FCS_CKM.1, FCS_CKM.3 and FCS_CKM.4 are used for authentication.
Security management: Only the MSM is authorized to load encrypted MSM Controls Data (FMT_MTD.1). Because of the encryption, an unauthorized user cannot modify this data. FPT_TDC.1, FMT_MOF.1, FMT_MSA.1, FMT_MSA.2 and FMT_MSA.3 are used as support.
Protection of the TSF: FPT_PHP.3 prevents unauthorized users modifying or corrupting sensitive information in memories using physical attack. FPT_TST.1 provides validity checks, when the MCD is powered-up or reset, to prevent unauthorized modification or corruption of data.
Security audit: FAU_ARP.1 and FAU_SAA.1 monitor problems related to corruption or unauthorized modification of sensitive information.

O.FLAW*
The objective is met by good design and testing as specified by EAL5 augmented conformity requirements.

O.CLON*
The protection against cloning objective is assured by the following:
Good key housekeeping: FCS_CKM.4 is a protection against unauthorized disclosure of cryptographic keys. Cryptographic keys had to be held securely to prevent cloning.
Unobservability of TSF data: FPR_UNO.1 prevents observation of TSF data, which is necessary for cloning the TOE.
Resistance to physical attacks: Physical attacks are able to determine some important data of the TOE (i.e. cryptographic keys) which is necessary for cloning. FPT_PHP.3 prevents that.
Other SFR also participate in cloning prevention:
Cryptographic support: FCS_CKM.1, FCS_CKM.3, FCS_COP.1

Data protection: FDP_ACC.2 and FDP.ACF.1 manage access to users' data, which must not be modified to prevent cloning. Indeed, a fake application can be used to access to sensitive data. FDP_DAU.1 is used to verify the validity of applications code spaces.
Identification and authentication: FIA_AFL.1, FIA_UAU.1, FIA_UID.1, FIA_UAU.4, FIA_USB.1, FMT_MOF.1 are used to prevent unauthorized users gaining access to sensitive data using identification and authentication mechanisms. It is a way to protect the TOE from cloning.
Security management: Management of security functions is assured by the one and only MSM that loads encrypted MSM Controls Data on the MCD.
MSM Controls Data contains security attributes that determine the behavior of some security functions. FMT_MSA.1 defines all of the security attributes which are loaded by the MSM Controls Data. The MSM encrypts the MSM Controls Data to ensure these are not tampered with, which provides FMT_MSA.2. Because the key used for this is the TK,V known only by the MSM, FMT_MTD.1 is achieved. The MSM has control over the generation of the MSM Controls Data which allows for FMT_MSA.3.

O.ROLLBACK
This objective is assured by the following:
The TOE is in a valid state before a loading of an application thanks to FPT_FLS.1 and FPT_RCV.4. In the case of a failure in the loading, FDP_ROL.1 allows the erasure of the application's code and data automatically.

O.RESOURCE
Resource preservation objective is assured by the following:
Management of limits on TSF data: FMT_MTD.2.
Maximum quotas: FRU_RSA.1.
Backed by FAU_ARP.1 Security Alarms and FAU_SAA.1 Potential violation analysis.

O.LOAD
The control of the administrator is assured by the following:
Identification and authentication: No load operation can be done before identification/authentication operation succeeds (FIA_UAU.1 and FIA_UID.1).
Security management: MSM is the only user authorized to load MSM Controls Data. MSM Controls Data contains data used to determine the behavior of the security functions used to manage the load and delete process of the applications (FMT_MSA.1, FMT_MSA.2, FMT_MSA.3, and FMT_SMR.1). FMT_MTD.2 also implements this objective.
User data protection: The complete access control (FDP_ACC.2 and FDP_ITC.1) of the load operations allows verification of the ALC or ADC that is produced by the MSM by comparing security attributes of the application and security attributes stored on the MCD.
Cryptographic support: FCS_COP.1.

O.SECURITY
Loading of applications requirements is assured by the following:
Import of User Data without Security Attributes: FDP_ITC.1.
Cryptographic support: FCS_COP.1, FCS_CKM.1, FCS_CKM.3 and FCS_CKM.4.

O.EFFECT_L
Separation of the Loaded-Applications is assured by the following:
Security attributes: The Unique Application Identifier identifies each application stored in the memory of the MCD (FDP_ACF.1). Indeed, each application is stored in an Application Pool Block (which contains its code and data) which is identified by the Unique Application Identifier.
And the following SFR which assure correct operation:
Failure with preservation of secure state: FPT_FLS.1.
Function recovery: Security functions involved in application load have the capacity to either complete or to recover to a consistent and secure state (FPT_RCV.4).

O.REMOVE
Safety of the removal is assured by the following:
Identification and authentication: No delete operation can be done before identification/authentication operation succeeds (FIA_UAU.1 and FIA_UID.1).

Security management: the MSM is the only user authorized to load MSM Controls Data. The MSM Controls Data contains data used to determine the behavior of the security functions used to manage the load and delete process of the applications (FMT_MSA.1, FMT_MSA.2, FMT_MSA.3, and FMT_SMR.1).
User data protection: The complete access control (FDP_ACC.2) of the load and delete operations permits the verification of the ALC or ADC that is produced by the MSM (the administrator) by comparing security attributes of the application and security attributes stored on the MCD. FDP_ITC.1 also implements this objective.
Cryptographic support: FCS_CKM.1, FCS_CKM.3, FCS_CKM.4 and FCS_COP.1.
Information protection: FDP_RIP.1 guarantees that the code and the application's data are erased and are no longer accessible after the removal.

O.EFFECT_R
Separation of the unloaded applications from the other Loaded-Applications is assured by the following:
Security attributes: The Unique Application Identifier permits the identification of each application stored in the memory of the MCD (FDP_ACF.1). Indeed, each application is stored in an Application Pool Block, (which contains its code and data) which is identified by the Unique Application Identifier.
And the following SFR which assures correct operation:
Failure with preservation of secure state: If an application attempts to modify code or data of remaining Loaded-Applications, a secure state is preserved (FPT_FLS.1).
Function recovery: Security functions involved in application deletion have the property to either complete or to recover to a consistent and secure state (FPT_RCV.4).

O.SEGREGATE
Segregation of Loaded-Applications is assured by the following:
User data protection and security management: The Unique Application Identifier, contained in the ALC (FDP_ACF.1), is used to identify the Application Pool Block in which the application's code and data are stored. It is directly coupled to the application load process but is also a basic requirement of segregation of Loaded-Applications (FDP_ACC.2).
The TSF fulfilling the SFR are protected by:
Failure with preservation of secure state: If a failure occurs (Loaded-Application trying to read from or write to another's Loaded-Application's code or data without authorization), a secure state is preserved (FPT_FLS.1).

O.CIPHER
This security objective is directly covered by FCS_CKM.1, FCS_CKM.3, FCS_CKM.4 and FCS_COP.1.

O.DECIPHER
This security objective is directly covered by FCS_COP.1/RSA.

7.5.2 SARs and the Security Requirements Rationale

7.5.2.1 Evaluation Assurance Level Rationale

An assurance requirement of EAL5 is required because the platform is designed to support Loaded-Applications intended to defend against sophisticated attacks.
This evaluation assurance level was selected since it is designed to permit a developer to gain maximum assurance from positive security engineering based on good commercial practices. EAL5 represents a high practical level of assurance expected for a future commercial grade product. In order to provide a meaningful level of assurance that the TOE provides an adequate level of defense against such attacks, the evaluators should have access to the low level design and source code.

7.5.2.2 Assurance Augmentations Rationale

Additional assurance requirements are also required due to the definition of the TOE and the intended security level to assure.

ALC_DVS.2 Sufficiency of security measures

Development security is concerned with physical, procedural, personnel and other technical measures that may be used in the development environment to protect the TOE.
This assurance component is a higher hierarchical component to EAL5 (only ALC_DVS.1 is found in EAL5). Due to the nature of the TOE, there is a need to justify the sufficiency of these procedures to protect the confidentiality and the integrity of the TOE.
ALC_DVS.2 has no dependencies.

AVA_VAN.5 Advanced methodical vulnerability analysis

Vulnerability analysis is an assessment to determine whether vulnerabilities identified, during the evaluation of the development and anticipated operation of the TOE or by other methods (e.g. by flaw hypotheses), could allow users to violate the SFRs.
This assurance component is a higher hierarchical component to EAL5 (only AVA_VAN.4 is found in EAL5).
AVA_VAN.4 has dependencies with ADV_ARC.1 Security architecture description, ADV_FSP.4 Complete functional specification, ADV_TDS.3 Basic modular design, ADV_IMP.1 Implementation representation of the TSF, AGD_OPE.1 Operational user guidance, and AGD_PRE.1 Preparative procedures.
These components are included in EAL5 and AVA_VAN.5, and so these dependencies are satisfied.

8. TARGET OF EVALUATION SUMMARY SPECIFICATION

The TOE summary specification describes how the TOE meets each SFR.

8.1 SECURITY FUNCTIONALITY

This following defines the TOE security functions. The *italic paragraph parts* correspond to *actions provided by the security functions* whereas the normal paragraph parts correspond to the context in which the security functions take place.
Note that cryptographic primitives are out of scope.
Table 3 shows how these security functions satisfy the TOE security functional requirements.

8.1.1 Application Load Certificate Control SF (SF1)

SF1 ensures that the MSM Controls Data has been loaded before loading any application. SF5 maintains a flag to indicate whether or not MSM Controls Data has been loaded successfully onto the MCD. This flag is contained in MULTOS security data.
SF1 authenticates an application load certificate as having been authorized by the MSM, using the MSM's KCK (kck_pk), prior to validating the loaded-application. SF1 calculates an asymmetric hash of the key header and compares it with the deciphered Key Certificate, using a RSA algorithm and kck_pk.
SF1 ensures an authorized application has appropriate permissions (MCD Issuer product identifier, MCD Issuer identifier, MCD enables dates, MCD number, four permissions field reserved for future use) before load is validated. In this way, SF1 checks the eight application's permissions against the eight MCD's permissions.
SF1 checks if an application, which has been previously loaded and then, deleted, is authorized by the MSM to be reloaded. The ALC contains a value that indicates if reloads of the application onto the same MCD are authorized. The value can be zero or a random number generated by the MSM.
A value of zero means that the MSM has authorized multiple reloads of the application.
SF1 ensures the application is not already loaded on the MCD. When an attempt is made to load the application, the AID (unique Application Identifier) contained in the ALC is checked against the AID associated with each application already loaded on the MCD. If a match is found, this indicates the application has already been loaded onto the MCD and the load attempt will fail.
When loading the ALU components in the Application Pool Block in FLASH memory, SF1 checks if there is enough space available. If it is not the case, SF1 returns an error.
If load application fails, SF1 ensures that the temporary loaded-application is erased on the next reset.
Permutational/probabilistic/cryptographic mechanisms used in this security function: RSA and asymmetric hash algorithms.

8.1.2 Application Delete Certificate Control SF (SF2)

SF2 authenticates the Application Delete Certificate with the key kck_pk that is stored in the MCD's ROM. The authentication is done through an asymmetric hash of the ADC and a comparison with the signature provided.
SF2 delete an application only after receiving and authenticating a valid application delete certificate. SF2 checks that the Application ID extracted from the certificate matches to a loaded application and that the permissions are correct (the delete process uses the same interpretation of permissions as the load process). Only after all these checks have passed will SF2 delete the application.
Permutational/probabilistic/cryptographic mechanisms used in this security function: RSA algorithm, asymmetric hash algorithm.

8.1.3 Unprotected/Protected Application Load Unit SF (SF3)

SF3 manages the Unprotected/Protected Application Load Unit which is composed of the Application Code (clear text copy of the application) and the Application Signature (for the protected ALU).
The protected ALU is used for application authentication.
If the ALC indicates that application authentication is required (it is optional), the application is authenticated by its application signature. *When application authentication is invoked, SF3 verifies the authenticity of the application. Once the application has been loaded onto the MCD, SF3 creates a digest of the application using the one-way hash function. SF3 then decrypts the application signature using the Application Provider's public key (ack_pk), which is*

contained within the ALC. ack_pk is certified by the MSM. The MSM signs ack_pk using the secret KCK (kck_sk). SF1 can verify the authenticity of ack_pk by decrypting it with the public KCK (kck_pk).
The hash digest from within decrypted signature is compared with the application digest generated by SF3. If they are equal, then the authenticity of the application is confirmed, since only the Application Provider could create the application signature and the Application Provider's public key is certified by the MSM.
If the decrypted signature does not match the application digest generated by SF3, application authentication fails and SF3 aborts the loading of this application..
Permutational/probabilistic/cryptographic mechanisms used in this security function: RSA algorithm, asymmetric hash algorithm.

8.1.4 Confidential Application Load Unit SF (SF4)

SF4 manages the Confidential Application Load Unit which is composed of the Application Code (clear text copy of the application), the Application Signature (for application authentication) and the Key Transformation Unit (for application confidentiality).
If application confidentiality is required (it is optional) SF4 allows to load applications which have protected areas of code or data. In order to protect the confidentiality of an application, the Application Provider is able to encrypt the relevant areas of the application using DES CBC or Triple DES CBC. The DES or Triple DES encryption key and descriptors for each of the encrypted areas are then encrypted using the public transport key (mkd_pk) of the MCD onto which the application is to be loaded. This information is placed into a KTU. The KTU is appended to the ALU. This ensures the confidentiality of sensitive parts of an application before it is loaded onto an MCD.
Once the application is loaded, SF4 allows the decryption of the protected areas of the application so that the application can be securely executed on the MCD.
Once SF1 has authenticated the Application Load Certificate, SF4 decrypts the KTU using the MCD-specific secret transport key (mkd_sk). SF4 uses the DES or Triple DES key recovered from the decrypted KTU to decrypt the protected application areas and complete the process of loading the application. This ensures that the protected application can be executed once it is safely loaded onto the MCD (where its confidentiality is protected by the Application Execution Management SF).
After decrypting the KTU, SF4 checks that the msm_cotnrols_data_dates and MCD Number specified in the KTU matches the target MCD. SF4 also checks that the application id specified in the KTU matches the application id in the ALC for this application before proceeding. If these details do not match, the application load attempt fails. This ensures that the presented KTU is intended for this application loading on to the MCD in question.
SF4 ensures only an authentic MCD is able to load and execute a protected application.
Since the MCD-specific secret transport key is required in order to decrypt the protected application areas, only the target MCD can gain access to those areas.
By successfully decrypting the KTU and recovering the DES or Triple DES key to decrypt the protected application areas, SF4 also authenticates the MCD as a valid MCD.
This ensures the confidentiality of the protected application in the event it is loaded onto a smartcard that is not an authentic MCD.
Permutational/probabilistic/cryptographic mechanisms used in this security function: RSA algorithm, DES algorithm.

8.1.5 MSM Controls Data Load Management SF (SF5)

During implementation of MULTOS in silicon for the target processor, MULTOS security data is injected into non-volatile memory. The MULTOS security data includes an MCD-unique identifier (the MCD id), an MCD-unique symmetric transport key (TKV) and a security flag.
MSM Controls Data for a specific MCD includes the MCD-unique identifier, the MCD's permissions and the MCD-unique transport key (MKD). MSM Controls Data is encrypted by the MSM using the MCD-unique symmetric transport key (TKV). MSM Controls Data is provided to the MCD Issuer for loading on the target MCD.
SF5 ensures the MCD only allow MSM Controls Data to be loaded once. The MCD Issuer presents the MSM Controls Data to the target MCD. This is done by submitting the Set MSM Controls command to MULTOS via an IFD. *Before loading, SF5 checks the security data flag to verify if the MSM Controls Data has not already been installed. If the MSM Controls Data have already been loaded, the attempt to load is denied.*
SF5 ensures encrypted MSM Controls Data is able to be loaded on target MCD. Since MSM Controls Data is encrypted using a symmetric key specific to the target MCD, only the target MCD is able to decrypt the data and load it successfully. Furthermore, the MCD is able to load only its own MSM Controls Data, since it will not be able to decrypt any other MCD's MSM Controls Data.

This ensures an MCD cannot load MSM Controls Data intended for another MCD and therefore cannot masquerade as another MCD (e.g., in order to load applications not intended for it).
SF5 verifies the integrity of the MSM Controls Data. SF5 generates a hash digest of the MSM Controls Data (less the last 16 bytes) and compares it with the attached hash digest (last 16 bytes of the decrypted data). If the two digests match, the MSM Controls Data has been received without corruption or tampering.
SF5 ensures the MCD only loads its own unique MSM Controls Data. If the decrypted MCD-unique identifier does not match the MCD-unique identifier stored in FLASH memoryof the targeted MCD, the MSM Controls Data is rejected.
If the MSM Controls Data is loaded successfully, SF5 sets the security flag to '0x5A' in MULTOS security data to indicate this has occurred. This ensures that once the MCD Issuer has enabled and issued the MCD, bogus MSM Controls Data cannot be created and loaded onto the MCD.
SF5 maintains a count of the number of failed (or incorrect) attempts to load the MSM Controls Data. SF10 monitors the value of this counter, and when the pre-determined limit is reached, the MCD shall be permanently shutdown from the next power on.
Permutational/ probabilistic/cryptographic mechanisms used in this security function: DES algorithm, asymmetric hash algorithm.

8.1.6 Application Execution Management SF (SF6)

SF6 ensures each application is restricted to accessing its own code and data. The only exceptions to the restriction on an application's code and data access are as follows:

a) Accessing data in the Public Data Area.

b) Application delegation.

c) Accessing Codelets.

d) Accessing data via MULTOS primitives.

SF6 also allows strong cryptography provided by the MCD to be regulated so that only authorized applications can access them.
SF6 maintains separate storage and execution space for applications loaded onto an MCD. SF6 manages a *pool of loaded applications. SF6 ensures each application, including its code and data areas, is kept separate from every other application loaded on the MCD. This ensures an application that is restricted to its own code and data space cannot gain access to the code or data of another loaded application.* Each application is allocated to its own Application Pool Block within the Application Pool. Each Application Pool Block contains a unique identifier of the application loaded into the block. The intent of this mechanism is to allocate a portion of FLASH memory to an application where that portion does not overlap regions allocated to any other applications and to tag these regions of memory with the application ID of that application.
An application is able to read code for execution only from its own code space or from a pool of common routines controlled by SF6. SF6 executes only applications written in the MULTOS Execution Language (MEL). MEL is an interpreted language. MEL applications are executed on an Application Abstract Machine, which enables memory accesses by applications to be checked at the time of interpretation. SF6 ensures any attempt by an application to access code for execution is restricted to its own code space or to Codelets, which are controlled by SF6. This ensures an application is unable to compromise the integrity or confidentiality of the code of other applications loaded on the MCD.
SF6 ensures no application is able to write to the code space of any application, including itself. SF6 ensures any attempt by an application to write data is restricted to the application's own data space or to the Public data area. Any attempt by the application to write data outside these areas, including to its own or another applications code space, is blocked by SF6 and the application is terminated.
SF6 ensures no application is able to read from or write to the data space of another application except via a mechanism provided and controlled by SF6. SF6 ensures any attempt by an application to read or write data is restricted to the application's data space or to the Public data area. The Public data area is available for reading and writing by all applications and provides the mechanism for applications to communicate information with each other. This ensures an application is unable to compromise the integrity or confidentiality of the data of other applications loaded on the MCD.
No application is able to cause the execution of another application except via a mechanism provided and controlled by SF6. SF6 also provides a mechanism for an application to delegate execution to another application. On delegation, a full context switch occurs, so the only information from the delegating application which is available to the delegated application is whatever might be held in the Public data area. (Full context switch means that SF6 writes all information

related to the execution of the delegating application to an area under its control, then commences execution of the delegated application. When the delegated application ends its execution, the execution context of the delegating application is restored and it is able to continue execution from the point of delegation.) occurs, so the only information from the delegating application which is available to the delegated application is whatever might be held in the Public data area.
This ensures an application cannot make use of another application to compromise the integrity or confidentiality of other applications. Applications execute only within their own environment and cannot be made to execute in another application's environment.
SF6 ensures no application is able to write to the code space of MULTOS and no application is able to read from or write to the data space of MULTOS except via a mechanism provided and controlled by SF6. SF6 provides system primitives that can be invoked by applications, which return to the application specific system data values and allow specific system data values to be updated. Any other attempt by an application to access MULTOS code or data is blocked by SF6. This ensures no application is able to compromise the integrity of MULTOS or the confidentiality of its sensitive information.
SF6 ensures only applications specifically authorized by the MSM can access strong cryptography primitives. The ALC contains a flag indicating whether or not the application is authorized to use MULTOS's strong cryptography primitives. This information is stored with the application when it is loaded onto the MCD. *Every time an application attempts to call a strong cryptography primitive, SF6 checks the control flag to determine if the application is allowed to make the call. If it is, SF6 will process the call. If the flag indicates access is not authorized, SF6 will return an error condition to the application.* The MSM wishes to control which applications can access strong cryptography. This is necessary to comply with government restrictions on the use by Application Writers of strong cryptography. An Application Writer must obtain appropriate documentation (e.g., an export license) from the appropriate government body before the MSM will authorize the application's use of strong cryptography. The MSM authorizes an application to use strong cryptography by digitally signing its ALC with the cryptography access flag set to allowed.
SF6 ensures a series of functions that allow MULTOS to address all required RAM and FLASH memory it needs as follows: MULTOS needs to be able to access data held in up to 448KB of FLASH memory/ 12KB of RAM.

8.1.7 Critical Data Overwrite SF (SF7)

SF7 ensures that no part of an application's code or data, excluding data the application has placed into the Public data area, can be accessed after the application has been deleted.
When SF7 deletes an application from the MCD, it overwrites the application's code and data spaces with a fixed pattern of bytes. In this way, any other application subsequently loaded into the same space will be unable to determine any information relating to the deleted application.
Data that the application has written to the Public data area is not overwritten, since this provides the means for the application to communicate with other applications. By placing data in the Public data area, an application is effectively deciding the data can be accessed by any application.

8.1.8 Reset Protection SF (SF8)

When allocating memory to an application, a number of pointers must be manipulated. These pointers are held in FLASH memory and are susceptible to corruption if the MCD should loose power while being updated. *To protect against this, SF8 establishes a critical region around the operations that update these pointers. If the MCD is powered down or reset while in this critical region, SF8 will permanently shutdown the MCD. In this way, SF8 ensures that critical memory allocation operations occur as an atomic operation (i.e., they are either not initiated or are guaranteed to complete).*

8.1.9 Integrity Checks SF (SF9)

SF9 protects MULTOS critical data by applying an integrity check to the following information:

a) MISA injected security data.

b) MSM Controls Data.

c) Application code spaces.

SF9 calculates a four-byte check sum over the MISA (MULTOS Injection Security Application) injected security data and the MSM controls data when the MSM controls have been successfully set. This check sum is re-calculated and

verified by SF9 before sending out a response to any command.
A failure of the check sum causes SF9 to abend the session. This integrity check hence ensures that the smartcard will not attempt to send any response that may be based on corrupted data.
In addition, when an application is loaded onto the MCD (by successful execution of the Create MEL Application command), SF9 calculates a four-byte check sum over the application's code space. SF9 stores this check sum in the application pool block for the application. When an application is selected as the current file, SF9 calculates the check sum over its code space and compares it with the stored check sum to confirm the continued integrity of the application. If the calculated and stored check sums do not match, SF9 abends the session.
A full integrity check verifies the checksum of the full MSM Controls data whereas a partial integrity check excludes the verification of the codelets within the MSM Controls data. A full integrity check is performed at startup and a partial integrity check is performed just prior to sending a response to every command in order to make sure that security data and MSM Controls data remain unchanged.
Therefore, MULTOS is not vulnerable to attempts to corrupt its memory.
Permutational/probabilistic/cryptographic mechanisms used in this security function: 4-byte checksum.

8.1.10 Start-up Validity Checks and Initialization SF (SF10)

If the MCD is reset or loses power while MULTOS is processing a command or executing an application, SF10 will perform the usual validity checks and initialization when MULTOS is restarted:

a) The MCD validity check allows SF10 to determine that MULTOS is still in a valid state (if it is not, SF10 will shutdown permanently).

b) SF10 erases the Public data area (to protect any sensitive information placed there by an application executing at the time of reset/power loss).

c) SF10 erases from the Application Pool any application in the Application Pool that is in the opened state (since the application load process has been interrupted).

d) SF10 initializes the Active Application Block to the shell application if any is present, or otherwise to a null value to indicate that no application is currently selected.

e) SF10 rolls back any uncommitted writes in the Data Item Buffer.

The validity check can fail for the following reasons:

a) *A check of the integrity of the security data (comprising Initialization Security Data and MSM Controls Data) fails, the MCD can no longer function correctly and SF10 abends.*
A change to the MCD's security data could indicate an attempt to attack the MCD or a failure of the MCD memory.

b) *The Application Memory Manager module detects it was in the middle of a critical operation when the system was reset. SF10 permanently shuts down the MCD* in this circumstance.
The Memory Manager Software Module's data will be inconsistent, with no means to recover it to a consistent state. Critical operations involve the manipulation of memory addresses associated with an application and cannot be recovered.

c) *The maximum number of failed attempts to execute the Set MSM Controls command has been reached; since MSM Controls Data cannot be successfully loaded, it is not possible to load applications, so SF10 permanently shuts down the MCD.* The decrementing of the counter forms part of SF5.

The Data Item Buffer or Data Item Stack holds a stack of data item copies. Each data item copy held in this stack contains a copy of a particular Static data item which MULTOS has, or is in the process of, updated as the result of executing an application MEL instruction or primitive.
This data item stack also contains information that allows SF10 to determine, for each data item copy in the stack, whether the source data item has been successfully and completely updated.
The data item copy contains the following items. These items are located within the data item copy in the order given, with the first item at the lowest address:

- Flags and byte counts that allow navigation through the data item stack to find the most recent data item copy, to create a new data item copy, or to determine whether the most recent data item copy is a copy of an item which is in the process of being updated.

- A pointer to the start of the data item which the data item copy refers to.
- A copy of the data item.

When a data item copy is created SF10 marks it as ACTIVE and when the source data item is successfully and completely updated SF10 marks the data item copy as USED. If the card is reset and SF10 founds an ACTIVE block on the stack, SF10 will copy it back to its original location and mark it as USED.
At the end of initialization, MULTOS is in the Ready state, waiting to process commands from the IFD. It therefore returns to a known secure state following a reset or power-down/power-up.
Permutational/probabilistic/cryptographic mechanisms used in this security function: 4-byte checksum.

8.1.11 Tamper Resistant Software Behaviors SF (SF11)

MisExecution Detection
When required, SF11 detects possible mis-executions of the operating system due to unexpected external electro-magnetic or mechanical interference. SF11 calculates a parameter in two independent ways. SF11 then compares the two results. If the two results do not match, SF11 will make the MCD become mute. By doing this, SF11 will always trap a single mis-execution of the code which causes one of the parameters to contain an incorrect value.
Failed Command Counter
This counter is a software counter-measure against power analysis.
To limit the number of allowed failed attacks; SF11 maintains a count of the number of unsuccessful attempts to perform critical operations that rely on cryptographic mechanism. It makes infeasible attacks on these operations that rely on brute force attacks on the underlying cryptographic mechanism that supports it. *SF11 uses this to protect the RSA decryption mechanism used when loading and deleting applications as well as to protect the DES decryption mechanism used during the loading of the MSM Controls data. If a sufficient number (20 attempts) of unsuccessful commands is presented for any of these operations, SF11 will take appropriate action for the operation in question.*
SF11 uses a down count that is initialized to its full value during manufacturing. SF11 decrements the counter before beginning the cryptographic mechanism that is being protected and re-increments it if the operation is successful.
Hardware Sensor Checks.
SF11 shall also perform any software checks of security sensors or features provided by the hardware.
The Infineon platform includes an Active Shield that can detect attempts to physically probe the chip. SF11 shall check the shield at regular intervals and cause the termination of the current session if the shield has been disturbed.
The Infineon platform includes a Current Scrambling Engine (CURSE) which attempts to mask the power consumption of the chip to external observation. SF11 shall initialize the CURSE on power up.
SF11 shall include a software check of the correct operation of the hardware Random Number Generator. The current session shall be ended if this check should fail.

8.1.12 Smartcard Authentication SF (SF12)

SF12 provides a means for MCD Issuers to determine that an MCD is an authentic initialized MCD prior to loading it with MSM Controls Data.
On request, SF12 provides a digest of the contents of a selected area of memory within an initialized MCD. For *that, SF12 uses the Check Data Command.* This digest can be used for comparison with the results of the same request applied to a known authentic initialized MCD, in order to verify the authenticity of the target MCD.
The digest had to be representative of the contents of the memory which is subject to authentication (i.e., the selected area of memory together with a fixed portion of MULTOS data). *So SF12incorporates a portion of fixed MULTOS data in the digest. SF12 requires as input:*
The start address of the memory area to be checked
The length of the memory area to be checked
A random challenge value.
SF12 performs a bit-wise exclusive OR function on the random challenge value and the first part of the fixed transport key (TKF). The result of this operation is concatenated, by SF12, with the second part of TKF and a one way hash algorithm applied to it. Using this hash as an initial value, SF12 computes a hash digest over the contents of the indicated memory area.
The inclusion in the digest of fixed MULTOS data (in the form of TKF) enables the authenticity of the MCD to be checked by comparing the digest with the result produced from the same request applied to the same area of memory on a known authentic initialized MCD. The random challenge value ensures the returned digest cannot be spoofed.

SF12 ensures it is not possible to infer from the digest any information regarding the contents of the memory area checked.
The digest is formed using a one-way hash function over the specified memory area and random challenge value. This acts to prevent any useful information being returned in the digest and therefore prevents any potential compromise of sensitive MULTOS information.
SF12 is only available on initialized MCDs (i.e., which have not yet been enabled).
This function is only useful for authenticating MCDs before they are enabled. Allowing its use after the MCD is enabled could provide a means for probing for information related to applications loaded on the MCD. As it serves no useful purpose once the MCD is enabled and, despite the way in which the digest is constructed, could be used to attack the MCD, it is prudent to disable this function after loading MSM Controls Data. If this function is called on an enabled MCD, an error condition is returned. No digest is calculated.
Permutational/probabilistic/cryptographic mechanisms used in this security function: RSA algorithm.

8.1.13 Application Programming Interface SF (SF13)

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

- TDES algorithm support 112-bit key and 168-bit key
- RSA algorithm supports up to 2048 bit keys (Std method or CRT method).
- AES algorithm with 128, 192 and 256 bit keys.
- Random generator uses the certified Hardware Random Generator that fulfils the requirements of AIS31 (see [ST_IC]).
- SHA-1, SHA224, SHA-256, SHA-384, and SHA-512 algorithms
- Diffie-Hellman based on exponentiation and on EC algorithm.
- PACE based on DH algorithm (integrated mapping and generic mapping)
- PACE based on ECDH algorithm (integrated mapping and generic mapping)

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 4096-bits long keys.
  - The TDES key generation (for session keys) uses the random generator.
  - AES key generation
  - DH key generation
  - ECDH key generation
- Key destruction: the TOE provides a specified cryptographic key destruction method that makes 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 IC security function (Cryptographic support) for Random Number Generator (see [ST_IC]).

RSA standard Key generation Algorithm - 1024,1536,2048	FCS_CKM.1
RSA CRT Key generation Algorithm - 1024,1536,2048, 4096	FCS_CKM.1
TDES Key generation Algorithm - 112	FCS_CKM.1
AES Key generation Algorithm - 128, 192, 256	FCS_CKM.1
ECC Key generation Algorithm - 160, 192, 224, 256, 320, 384, 512, 521	FCS_CKM.1
EC Diffie-Hellman Key agreement Algorithm 160, 192, 224, 256, 320, 384, 512, 521	FCS_CKM.1
DH Key agreement Algorithm 1024, 1280,1536, 2048	FCS_CKM.1
Key access	FCS_CKM.3
Key destruction	FCS_CKM.4
RSA standard Signature & Verification – RSA SHA PKCS#1, RSA SHA PKCS#1 PSS – 1024,1152,1280,1536,2048	FCS_COP.1
RSA CRT Signature & Verification – RSA SHA PKCS#1, RSA SHA PKCS#1 PSS 1024,1152,1280,1536,2048, 3072, 4096	FCS_COP.1
RSA standard Encryption & Decryption – 1536, 1792, 2048	FCS_COP.1
RSA CRT Encryption & Decryption – 1024,1152,1280,1536,2048, 3072, 4096	FCS_COP.1

TDES Encryption & Decryption – DES NOPAD, DES PKCS#5, DES 9797 M1 M2 – 112, 168	FCS_COP.1
TDES Signature & Verification – DES MAC ISO9797-1 M1 M2, DES MAC NOPAD, DES MAC PKCS#5- 112, 168	FCS_COP.1
AES Encryption & Decryption – AES 128 NOPAD – 128, 192, 256	FCS_COP.1
AES Signature & Verification – AES MAC 128 NOPAD – 128, 192, 256	FCS_COP.1
ECDSA Signature & Verification – ECDSA SHA – 160, 192, 224, 256, 320, 384, 512, 521	FCS_COP.1
SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 Message digest	FCS_COP.1
ECC for PACE Integrited Mapping & Generic Mapping 160, 192, 224, 256, 320, 384, 512, 521	FCS_COP.1
DH for PACE Integrited Mapping & Generic Mapping 1024, 2048	FCS_COP.1
ECC for Pseudonym signature 160, 192, 224, 256, 320, 384, 512, 521	FCS_COP.1

Glossary

Abbreviations and Acronyms

Term	Description
ABEND	Abnormal End (of MEL application execution).
ADC	Application Delete Certificate.
ALC	Application Load Certificate.
ALU	Application Load Unit.
APB	Application Pool Block.
ATR	Answer To Reset.
CC	Common Criteria (for Information Technology Security Evaluation, Version 2.1).
CM	Configuration Management.
DES	Data Encryption Standard (algorithm).
EAL	Evaluation Assurance Level.
ES	Embedded Software
IC	Integrated Circuit.
IFD	Interface Device (to smartcard).
IT	Information Technology.
KTU	Key Transform Unit.
MAOSCO	MAOSCO refers to the MULTOS Consortium. The MULTOS Consortium controls the MULTOS specification and is responsible for advancing the MULTOS OS in all smartcard related markets.
MCD	MULTOS Carrier Device.
MEL	MULTOS Executable Language (application language).
MSM	MULTOS Security Manager.
OSP	Organizational Security Policies.
PP	Protection Profile.
RAM	Random Access Memory.
ROM	Read Only Memory.
RSA	Rivest-Shamir-Aldeman (algorithm).
SAR	Security Assurance Requirement.
SFR	Security Functional Requirement.
SFP	Security Function Policy.
ST	Security Target.
TOE	Target Of Evaluation.
TSC	TSF Scope of Control.

Term	Description
TSF	TOE Security Functions.
TSFI	TSF Interface.
TSP	TOE Security Policy.

Vocabulary

Term	Description
Embedded software	Software embedded in a smartcard IC. Embedded software may be in any part of the FLASH memoryof the IC.
Integrated circuit (IC)	Electronic component(s) designed to perform processing and/or memory functions.
Phases	Refers to the seven phases of the smartcard product lifecycle, as outlined in the Smartcard Integrated Circuit Protection Profile.
I/O peripherals	Material components of the TOE that manage its inputs/outputs.
Smartcard	A card according to ISO 7816 requirements, which has a non-volatile, memory and a processing unit embedded within it.
Smartcard embedded software	Composed of embedded software in charge of generic functions of the smartcard IC such as operating system, general routines and interpreters (smartcard basic software) and embedded software dedicated to the applications (smartcard application software).