Security Target

Virtual Machine
of
ID Motion V1

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 SECURITY TARGET REFERENCE

Security Target Title: Virtual Machine of ID Motion V1
Security Target Version Number: 1.4
ITSEF: Thales CEACI
Certification Body: ANSSI
Evaluation scheme: French

1.2 TOE REFERENCE

The Target of Evaluation is the **Virtual Machine** (VM) of ID Motion V1 platform mask on SLE78 family component. This is a Multos OS V4.3.1 based mask described as follows:

- **ML3-76**: MULTOS M3 masked on the Infineon SLE78CXxxxP IC with patch 0122v001 (i.e. SLE78CX1600P, SLE78CX1440P, SLE78CX1280P, SLE78CX800P, SLE78CX480P and SLE78CX360P). These ICs will be referred to as SLE78CXxxxP in the remainder of the document.

The SLE78CXxxxP IC (with product code M7801 A12) is certified in the BSI reports BSI-DSZ-CC-0606-2010 (reassessment 17 May 2011).

1.3 VERSION OF THE TOE

Each mask reference is identified by having an ic_type field.

- G231 mask on SLE78CX1600P (ML3-76) **is identified by having an** ic-type field value is: 0x76 and AMDID field of 0122v001

The ic_type field is returned as part of the response to the “Get Manufacturer Data” command, and the AMDID is returned as part of the response to the “Get Configuration Data” Command.

1.4 SECURITY TARGET OVERVIEW

The integrated circuit card (ICC), or smartcard, 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.
- Securely load many applications - applications should be able to co-exist on the smartcard.
- Ensure that applications are securely 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.

This security target focuses on the virtual machine that ensures the **secure execution** and the **segregation** of the applications (during **loading/deleting time** and **runtime**).

1.5 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.

The target of evaluation, the Virtual Machine made of the Abstract Application Machine and the Application Memory Manager, is part of of MULTOS OS (see Figure 1), embedded on the product identified in section 1.1. More precisely, the present TOE is a subset of the TOE described in [ST_full_IDMotion] and embedded into the ID Motion MULTOS V4.3 product. Therefore, the product description, usage and life cycle are described in the corresponding security target [ST_full_IDMotion] evaluated at the EAL5+ level.

1.5.1 TOE boundaries

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 MULTOS operating system is decomposed into fourteen (14) modules (equivalent to the term “subsystems” used by Common Criteria). Figure 1 shows these modules using a layered model, with the MULTOS operating system modules shaded. Each layer requests services from lower layers and provides services to higher layers. Each subsystem is designed to encapsulate and hide the data that it owns.

The hardware-dependent subsystems consist of
- RF Lib : Radio Frequency Library
- HW: Hardware Services subsystem
- Crypto Libs: Cryptographic Libraries

The MULTOS-independent subsystems consist of
- SS: Startup Shutdown subsystem
- IO: I/O Communications subsystem
- CF: Cryptographic Functions subsystem
- UF: Utility Functions subsystem

The Initialization Mode subsystem consists of
- IM: Initialization Mode subsystem

The MULTOS-specific subsystems consist of
- CR: Command Router subsystem
- CH: Command Handlers subsystem
- MM: Application Memory Manager subsystem
- SD: System Data subsystem
- MULTOS API: MULTOS API subsystem
- Native Libs: Native Libraries
- AM: Application Abstract Machine subsystem

The TOE is the Virtual Machine that is composed of the AM and the MM subsystems.
- The AM subsystem ensures the interpretation of the MULTOS applications (in runtime)
- The MM subsystem ensures the management of the memory space dedicated to each application. In particular, the MM allocates this space in loading time and controls the access to its elements in runtime.

Beside the TOE, the product also contains the other subsystems of the MULTOS OS and some following ROMed applications (these elements are out of scope of the TOE):

- **Mel applications and modules ROMed**:
  - MPCOS V3.7
  - Pin Server Application (PSA) v1.0

  - Etravel EAC v1.4
  - IAS Classic v3.5
- **Mel applications in EEPROM**:
  - MOCA client v1.0

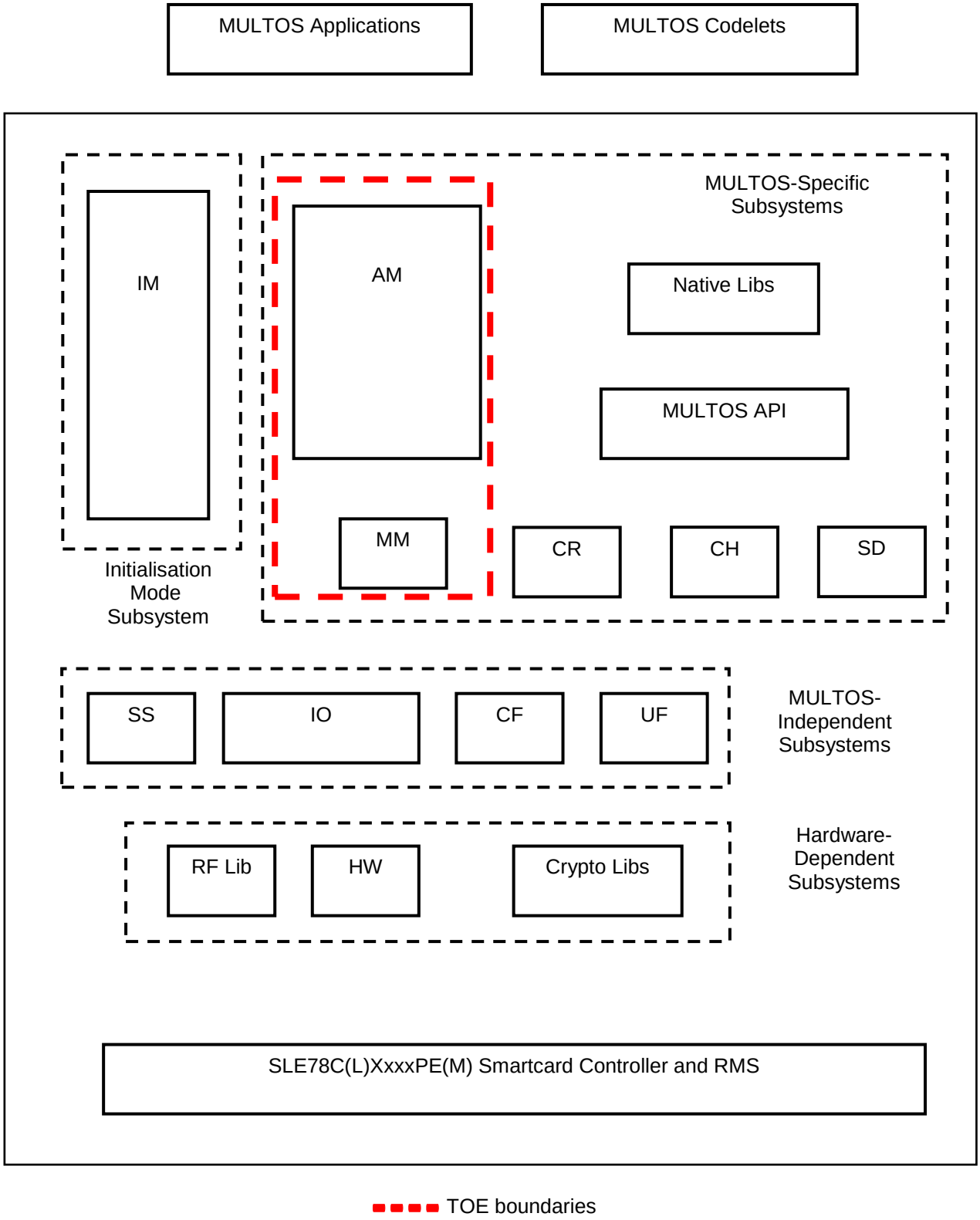

Figure 1: TOE inside the layered structure of MULTOS OS

1.5.2 TOE functions

The two subsystems of the TOE (namely AM and MM) implement the following functions:

1. Managing the memory effects of the application life-cycle i.e. opening, loading, creation, selection, de-selection, exit and removal: the control of load and removal certificates/permissions is however not ensured by the TOE but by the CH (Command Handlers) subsystem.
2. Interpreting the MEL instructions and primitives contained in the loaded applications
3. Handling the interaction between loaded applications (by delegation mechanism)

The TOE functions enforce the access control policy of MULTOS. Indeed, the TOE maintains separate storage and execution space for applications loaded onto an MCD (MULTOS Carrier Device) (using function 1). The application execution mechanism (i.e. the functions 2 and 3) ensures that each application, including its code and data areas, is kept separate from other loaded applications. Each loaded application is restricted to its own code and data space and cannot gain un-authorized access to the code or data of another loaded application.

1.5.3 Product usage

With respect to the off-card entities, the TOE is seen as part of the MULTOS operating system and hence, the TOE usage is the one of the TOE of the ID Motion MULTOS V4.3 product, described in [ST_full_IDMotion].

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 onto 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 one can be called by an executing application. Codelets can be loaded into MULTOS during IC manufacturing 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 SLE78C(L)XxxxP(M) Smartcard Integrated Circuits (ICs). The IC provides the microprocessor that executes the instructions including the executable code of MULTOS. The Infineon Technologies SLE78CXxxxP is a contact interface integrated circuit (see Hardware reference manual for details [HW-Manual]).

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.

Note that cryptographic primitives are out of scope.

1.5.4 Smartcard Product Life Cycle

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

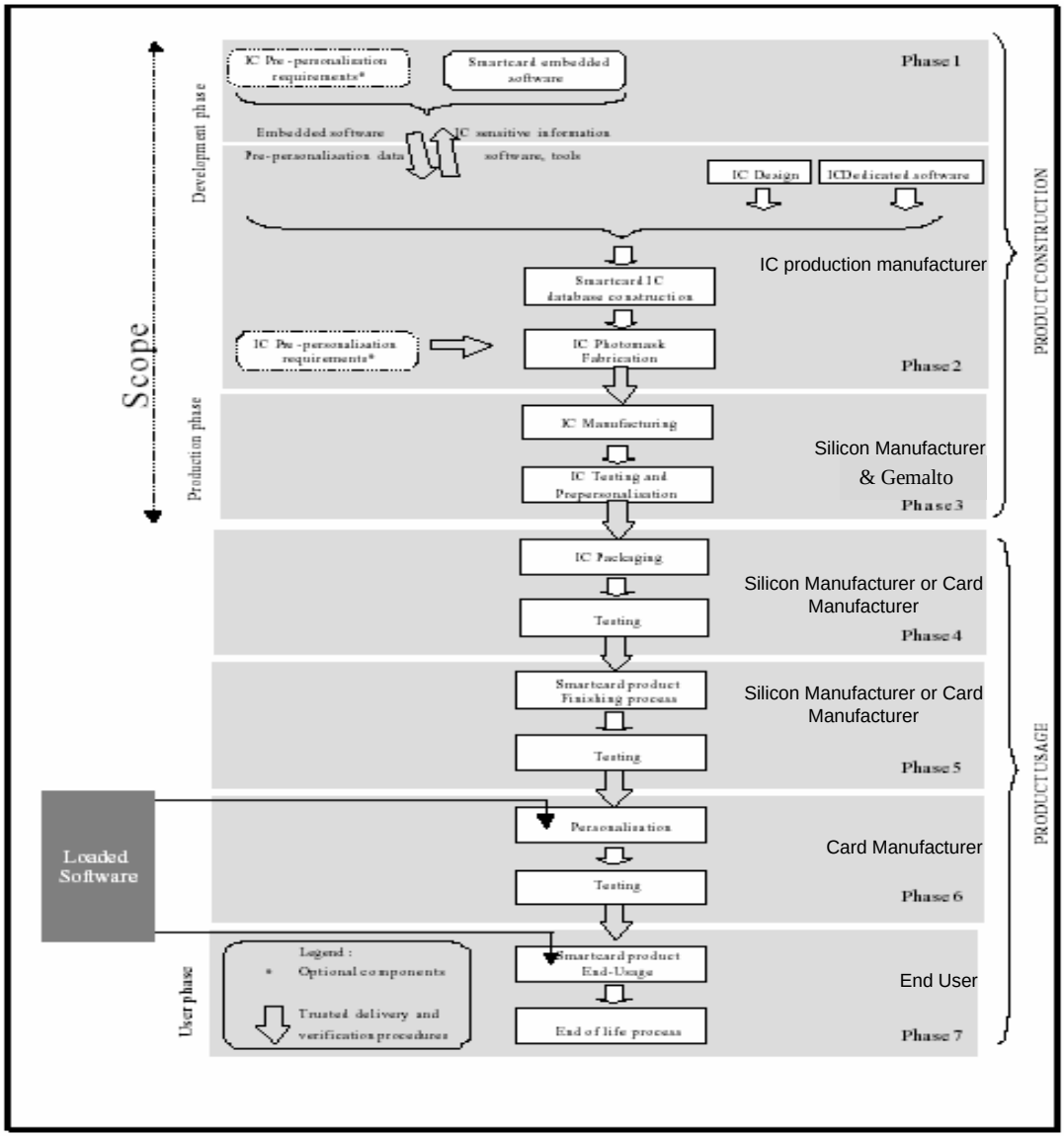

Figure 2: Smartcard IC with Multi-Application Platform Life Cycle

1.5.5 Product Location and Usage

MULTOS will initially be developed in software. Following successful implementation and testing, the MULTOS executable will be masked in Read Only Memory (ROM) and embedded on smartcards.
Once the MULTOS chip has been embedded on a target smartcard, interaction with it will be via commands issued to the card from an IFD or service requests (i.e., MULTOS system calls, known as primitives) made by an executing application.

1.5.6 Supporting Firmware

MULTOS requires firmware run-time libraries to support writing data to EEPROM. These libraries are supplied by Infineon Technologies. They provide low-level routines to support writing data to EEPROM, 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 EEPROM.

1.5.7 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 2 provides a simplified context diagram of the MULTOS commercial framework and security infrastructure.

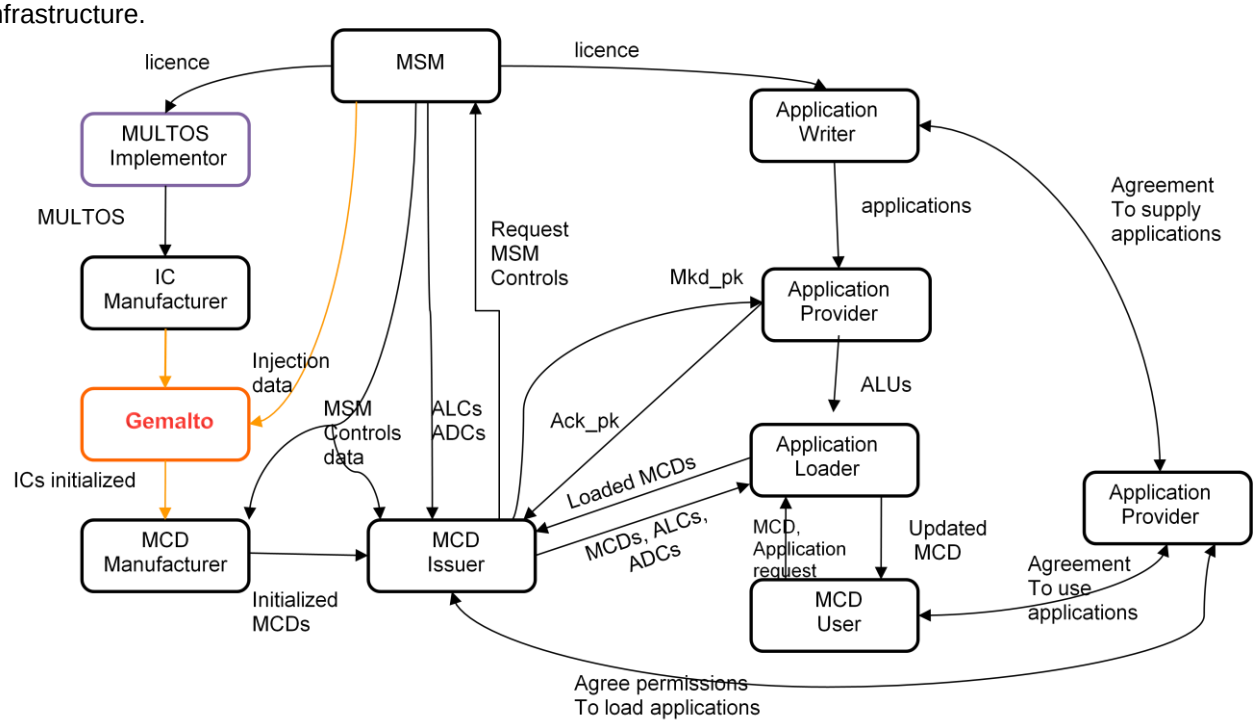

Figure 3: MULTOS Infrastructure Context Diagram

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 EEPROM injection including a diversified transport key (used by Initialization Mode).
- Gemalto then performs the final manufacturing step by injecting the iKMA keys and data into the EEPROM.

.

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 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 EEPROM injection,
- 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 EEPROM;

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.

f) **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.

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

h) **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.

i) **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.

j) **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.

k) **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.
MA-cards 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.

1.5.8 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.

1.5.9 Key Transformation Unit (KTU)

An Application Provider wishing to use 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.

1.5.10 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 certification 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. For the initial version of MULTOS, the only regulated features are strong cryptography functions.
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.

1.5.11 Keys

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

Key Name	**Type**	**Role**
kck kck_sk kck_pk	asymmetric	Global Key Certification Key Held securely by MSM ; used by MSM to certify ADCs and ALCs (and indirectly through these, Application Provider public keys (ack_pk)) Held in EEPROM of every MCD ; used by MULTOS to verify ALCs, ADCs and Application Provider public keys.
ack ack_sk ack_pk	asymmetric	Application Provider's asymmetric key; generated by Application Provider Held by Application Provider; used by Application Provider to sign application certificate. Provided to MCD Issuer, who gets it certified by the MSM when ALCs and ADCs are requested.

Key Name	Type	Role
tkck tkck_sk tkck_pk	asymmetric	Transport Key Certification Key Held securely by MSM; used by MSM to certify MCD-specific public transport keys (mkd_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).
tkv	symmetric	MCD-specific transport key; generated by MSM; stored in non-volatile memory of target MCD; used by MSM to encrypt MCD-specific MSM Controls Data and also by MULTOS to decrypt the MSM Controls Data.
mkd mkd_sk mkd_pk mkd_pk_c	asymmetric	MCD-specific asymmetric transport key. Held in non-volatile memory of target MCD; used by MULTOS to decrypt KTU. Held by MSM; stored in non-volatile memory of target MCD; copy provided to Application Providers; used by Application Providers to encrypt KTU for target MCD. mkd_pk, certified by MSM using tkck_sk to indicate it's authenticity. By decrypting this with tkck_pk the mkd_pk can be recovered for use.
tkf	symmetric	Fixed part of MCD-specific transport key; generated by MSM; stored in non-volatile memory of MCD; used by MSM, MCD Issuer and Application Loader to check authenticity of target MCD; tkf is fixed for all MCDs.
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.
hm	asymmetric	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.

Table 1: 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 (GKCK) (identified above as kck).

b) Transport Key Certification Key (TKCK) (identified above as tkck).

The GKCK supports the authentication of MULTOS applications and the authorization of requests by MCD Issuers to load and delete applications. The secret GKCK (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 GKCK (kck_pk) is installed in the ROM Mask 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 utilise

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.

1.5.12 MULTOS Initialization Security Data

MULTOS Initialization Security Data is generated by the MSM and supplied to Gemalto for incorporation into MULTOS. MULTOS Initialization Security Data comprises two elements:

a) The public GKCK (kck_pk) and Hash Modulus (hm), which is included in MULTOS EEPROM.

b) Security data, which is injected into non-volatile memory.

The MULTOS security data is injected into non-volatile memory 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.
The MISA then constructs the data to be injected into the MCD. The MULTOS security data includes:

a) A unique identifier based on the MISA identifier and ICC serial number.

b) 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.

c) Initialization date, indicating when the security data was injected into the MCD.

d) A security flag indicating MSM Controls Data has not been loaded.

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

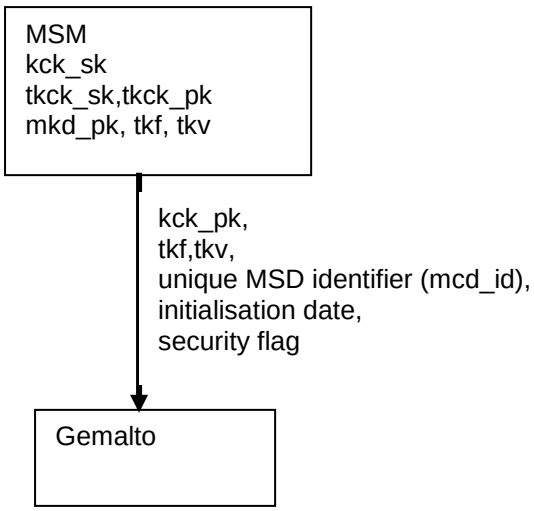

Figure 4: MULTOS Initialization Security Data Information Flow

1.5.13 MSM Controls Data

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.

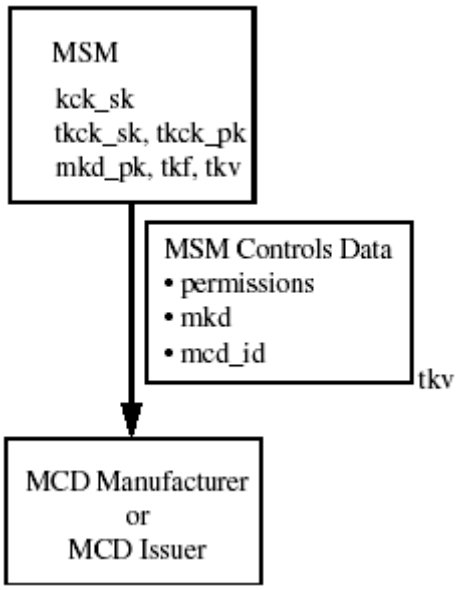

Figure 5: MSM Controls Data Information Flow

1.5.14 Loading Applications

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

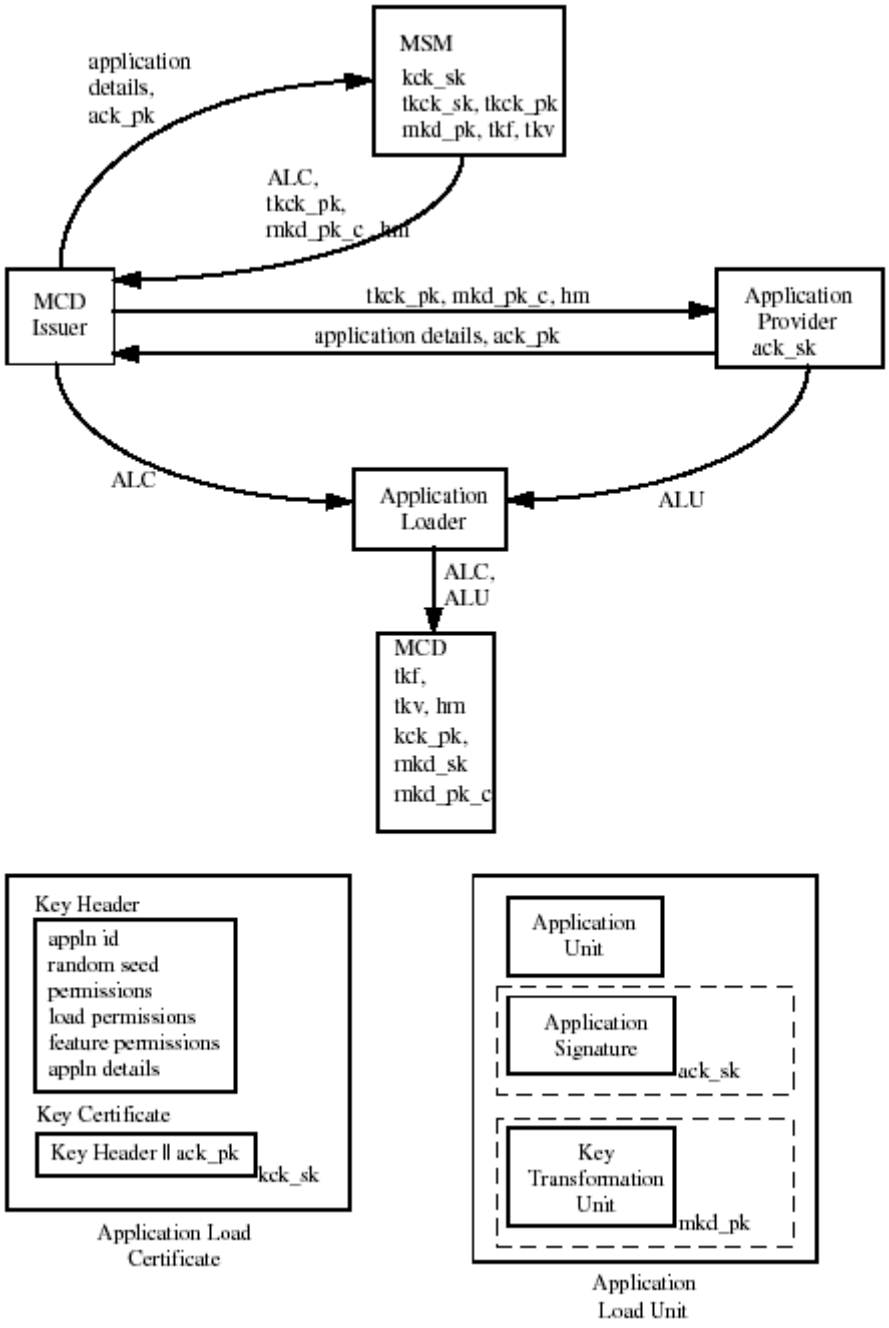

Figure 6 : Principal Key and Data Exchanges in Loading MCD Applications

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 GKCK (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 MSM 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 authorised by the MSM.

2. CONFORMANCE CLAIMS

This section describes how the ST claims conformance with Common Criteria for Information Technology Security Evaluation v3.1, revision 3.

2.1 COMMON CRITERIA CONFORMANCE CLAIMS

This ST has been built with Common Criteria for Information Technology Security Evaluation, Version 3.1, revision 3, as the following:

- Part 2 conformant.
- Part 3 conformant with EAL7 level.

2.2 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.

3. 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.

3.1 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 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 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.

3.2 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).

3.2.1 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.

3.2.2 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).

3.2.3 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 Personaliser.
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 Personaliser.

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 Personaliser

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 Personaliser.

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 Personaliser.

3.2.4 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.

3.2.5 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 load an application on the platform by 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 load 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.
Note: T.APP_DISC is also present during phase A1.

T.APP_CORR
Attackers load an application that partially or completely overwrites another Loaded-Applications, either corrupts or gains access to code or data. This is an attack on Application Provider user data.

T.APP_REMOVE
Attackers remove 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 remove 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 use 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 use 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 target total or partial destruction of card resources delivered by the platform.

3.2.6 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 2: Relationship between phases and threats

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

3.3 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 applets 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).

3.4 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.

3.4.1 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.

3.4.2 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).

3.4.3 Assumption on Phase 7

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

3.4.4 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.

3.4.5 Additional assumptions

The following assumptions are added because the corresponding TOE security objectives in [ST_full_IDMotion] is transferred to the TOE environment in this ST.

A.TAMPER_ES
Tampering with the security critical parts is prevented by the security mechanisms (in particular the unauthorized change of functional parameters, security attributes and secrets such as the lifecycle sequence flags and cryptographic keys).

A.SIDE
The interpretation of electrical signals from the hardware part is avoided.

A.CLON
The product functionality must be protected from cloning.

A.OPERATE
The continued correct operation of the security functions is assumed.

A.DIS_MECHANISM2
The embedded system security mechanisms are protected against unauthorized disclosure.

A.DIS_MEMORY
The sensitive information stored in memories is protected against unauthorized disclosure.

A.MOD_MEMORY
The sensitive information stored in memories is protected against corruption or unauthorized modification.

A.LOAD
Only application with permission of the administrator is loaded onto the platform.

A.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.

A.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.

A.CIPHER
There is a means to cipher sensitive data for application in a secure way. Supported cryptographic algorithms must be consistent with cryptographic usage policies and standards.

A.DECIPHER
There is a means to decipher the KTUs.

3.5 COMPOSITION TASKS

3.5.1 Statement of Compatibility – Threats

The following table lists the relevant threats of the SLE78CXxxxP and Derivates Security Target-lite relating to the IC product certified by the BSI under BSI-DSZ-CC-0606-2010 and provides the link to the threats related to the composite product, showing that there is no contradiction between the two.

IC Relevant Threat Label	IC Relevant Threat Title	IC Relevant Threat Content	Link to the composite-product threats
T.Phys-Manipulation	Physical Manipulation	An attacker may physically modify the Security IC in order to(i)modify User Data (ii) modify the Security IC Embedded Software (iii) modify or deactivate security services of the TOE, or (iv) modify security mechanisms of the TOE to enable attacks disclosing or manipulating the User Data or the Security IC Embedded Software.	T.DIS_ES2 T.T_CMD T.MOD_TSF T.MOD_EXE T.LOAD_MAN T.LOAD_APP T.APP_CORR T.APP_REMOVE T.ERR_REMOVE T.DEL_REMOVE T.APP_READ T.APP_MOD T.DIS_DATA T.MOD_DATA
T.Phys-Probing	Physical Probing	An attacker may perform physical probing of the TOE in order: (i) to disclose User Data (ii) to disclose/reconstruct the Security IC Embedded Software or (iii) to disclose other critical information about the operation of the TOE to enable attacks disclosing or manipulating the User Data or the Security IC Embedded Software.	T.DIS_ES2 T.MOD_TSF T.MOD_SOFT* T.DIS_DATA
T.Malfunction	Malfunction due to Environmental Stress	An attacker may cause a malfunction of TSF or of the Security IC Embedded Software by applying environmental stress in order to (i) modify security services of the TOE or (ii) modify functions of the Security IC Embedded Software (iii) deactivate or affect security mechanisms of the TOE to enable attacks disclosing or manipulating the User Data or the Security IC Embedded Software. This may be achieved by operating the Security IC outside the normal operating conditions.	T.DIS_ES2 T.T_CMD T.MOD_TSF T.MOD_EXE T.LOAD_MAN T.LOAD_APP T.APP_CORR T.APP_REMOVE T.ERR_REMOVE T.DEL_REMOVE T.APP_READ T.APP_MOD T.DIS_DATA T.MOD_DATA
T.Leak-Inherent	Inherent Information Leakage	An attacker may exploit information which is leaked from the TOE during usage of the Security IC in order to disclose confidential User Data as part of the assets. No direct contact with the Security IC internals is required here. Leakage may occur through emanations, variations in power consumption, I/O characteristics, clock frequency, or by changes in processing time requirements.	T.DIS_ES2 T.DIS_DATA
T.Leak-Forced	Forced Information Leakage	An attacker may exploit information which is leaked from the TOE during usage of the Security IC in order to disclose confidential User Data as part of the assets even if the information leakage is not inherent but caused by the attacker.	T.DIS_ES2 T.DIS_DATA
T.Abuse-Func	Abuse of Functionality	An attacker may use functions of the TOE which may not be used after TOE Delivery (e.g. test features) in order to: (i) disclose or manipulate User Data (ii) manipulate (explore, bypass, deactivate or change) security services of the TOE or (iii)	T.DIS_ES2 T.MOD_SOFT* T.APP_CORR T.APP_REMOVE T.ERR_REMOVE T.DEL_REMOVE

IC Relevant Threat Label	IC Relevant Threat Title	IC Relevant Threat Content	Link to the composite-product threats
		manipulate (explore, bypass, deactivate or change) functions of the Security IC Embedded Software or (iv) enable an attack disclosing or manipulating the User Data or the Security IC Embedded Software.	T.APP_READ T.APP_MOD T.DIS_DATA T.MOD_DATA

Note. Both T.DIS_ES1 and T.CLON* relate to Phase 1 which is before the existence of the IC product and T.RND is irrelevant to the composite product. Therefore, for this evaluation, all three of these threats are considered outside the scope of the TOE.

3.5.2 Statement of Compatibility – OSPs

The following table lists the relevant OSPs of the SLE78CXxxxP and Derivates Security Target-lite relating to the IC product certified by the BSI under BSI-DSZ-CC-0606-2010 and provides the link to the OSPs related to the composite product, showing that there is no contradiction between the two.

IC OSP Label	IC OSP Content	Link to the composite-product
P.Process-TOE	Protection during TOE Development and Production: An accurate identification is established for the TOE. This requires that each instantiation of the TOE carries this unique identification.	No contradiction with the present evaluation OSPs. Current evaluation has objectives related to delivery to IC manufacturer such as O.DLV_DATA, O.SOFT_DLV* and O.DLV_PROTECT*
P.Add-Functions	Additional Specific Security Functionality: The TOE shall provide the following specific security functionality to the Security IC Embedded Software: Data Encryptions Standard (DES) Triple Data Encryptions Standard (3DES).	No contradiction with the present evaluation. Present evaluation makes use of DES and RSA functions.

3.5.3 Statement of Compatibility – Assumptions

The following table lists the relevant assumptions of the SLE78CXxxxP and Derivates Security Target-lite relating to the IC product certified by the BSI under BSI-DSZ-CC-0606-2010 and provides the link to the assumptions related to the composite product, showing that there is no contradiction between the two.

IC assumption label	IC assumption title	IC assumption content	IrPA	CfPA	SgPA	Link to the composite product
A.Process-Sec-IC	Protection during Packaging, Finishing and Personalisation	It is assumed that security procedures are used after delivery of the TOE by the TOE Manufacturer up to delivery to the end consumer 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 means that the Phases after TOE Delivery are assumed to be protected appropriately.			X	A.DLV_PROTECT* A.DLV_AUDIT* A.DLV_RESP* A.USE-TEST* A.USE_PROD* A.USE-DIAG* The assumptions are the same.

IC assumption label	IC assumption title	IC assumption content	IrPA	CfPA	SgPA	Link to the composite product
A.Plat-Appl	Usage of Hardware Platform	The Security IC Embedded Software is designed so that the requirements from the following documents are met: (i) TOE guidance documents (refer to the Common Criteria assurance class AGD) such as the hardware data sheet, and the hardware application notes, and (ii) findings of the TOE evaluation reports relevant for the Security IC Embedded Software as documented in the certification report.		X		Fulfilled by the composite-SAR ADV_COMP.1 The Smartcard ES is designed so that the requirements are met.
A.Resp-Appl	Treatment of User Data	All User Data are owned by Security IC Embedded Software. Therefore, it must be assumed that security relevant User Data (especially cryptographic keys) are treated by the Security IC Embedded Software as defined for its specific application context.		X		Assets have to be protected in terms of confidentiality, authenticity and control of their origin. Assets linked to: • IC and MAP specific data • MULTOS Initialization Security Data • MSM Controls Data • NA TSF data := keys and identification data • Users of Loaded Applications • Loaded Application software • Loaded Application SF data The TOE owns security relevant User Data. A.DIS_MEMORY* A.MOD_MEMORY* A.LOAD A.REMOVE A.SECURITY O.SEGREGATE

IC assumption label	IC assumption title	IC assumption content	IrPA	CfPA	SgPA	Link to the composite product
A.Key-Function	Usage of Key-dependent Functions	Key-dependent functions, if any, shall be implemented in the Security IC Embedded Software in a way that they are not susceptible to leakage attacks (as described under BSI.T.Leak-Inherent and BSI.T.Leak-Forced). Note that here the routines that may compromise keys when being executed are part of the Security IC Embedded Software. In contrast to this the threats BSI.T.Leak-Inherent and BSI.T.Leak-Forced address (i) the cryptographic routines which are part of the TOE and (ii) the processing of User Data including cryptographic keys.		X		A.TAMPER_ES A.SIDE O.DEV_DIS_ES O.INIT_ACS

3.5.4 Statement of Compatibility – Security Objectives

The following table lists the relevant security objectives of the SLE78CXxxxP and Derivates Security Target-lite relating to the IC product certified by the BSI under BSI-DSZ-CC-0606-2010 and provides the link to the security objectives related to the composite product, showing that there is no contradiction between the two.

IC objective label	IC objective title	Link to the composite-product
O.Phys-Manipulation	Protection against Physical Manipulation	O.TAMPER_ES O.SIDE O.CLON* O.FLAW* O.DIS_MECHANISM2 O.DIS_MEMORY*
O.Phys-Probing	Protection against Physical Probing	O.TAMPER_ES O.SIDE O.CLON*
O.Malfunction	Protection against Malfunction due to Environmental Stress	O.TAMPER_ES O.SIDE
O.Leak-Inherent	Protection against Inherent Information Leakage	O.MOD_MEMORY* O.SIDE
O.Leak-Forced	Protection against Forced Information Leakage	O.DIS_MEMORY* O.SIDE
O.Abuse-Func	Protection against Abuse of Functionality	O.TAMPER_ES O.SIDE O.OPERATE* O.DIS_MECHANISM2
O.Identification	TOE Identification	No contradiction with the present evaluation.
O.RND	Random Numbers	No contradiction with the present evaluation.

IC objective label	IC objective title	Link to the composite-product
O.Add-Functions	Additional specific security functionality	No contradiction with the present evaluation. Present evaluation makes use of DES and RSA functions.
OE.Plat-Appl	Usage of Hardware Platform	The Smartcard ES is designed so that the requirements are met. No contradiction.
OE.Resp-Appl	Treatment of User Data	The OS owns security relevant User Data. O.DIS_MECHANISM2 O.DIS_MEMORY* O.MOD_MEMORY* O. RESOURCE O.LOAD O.SECURITY O.EFFECT_LOAD O.REMOVE O.EFFECT_REMOVE O.SEGREGATE
OE.Process-TOE	Protection during TOE Development and Production	No contradiction with the present evaluation. 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.DLV_DATA O.TEST_OPERATE* O.USE_DIAG* O.APPLI_DEV
OE.Process- Sec-IC	Protection during Packaging, Finishing and Personalization	No contradiction with the present evaluation. 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.DLV_DATA O.TEST_OPERATE* O.USE_DIAG* O.APPLI_DEV

3.5.5 Statement of Compatibility – SFRs

The following table lists the relevant SFRs of the SLE78CXxxxP and Derivates Security Target-lite relating to the IC product certified by the BSI under BSI-DSZ-CC-0606-2010 and provides the link to the SFRs related to the composite product, showing that there is no contradiction between the two.

Security Functional Requirement	Refined in [PP]	Link to the composite-product
FRU_FLT.2 "Limited fault tolerance"	Yes	No conflict.
FPT_FLS.1 "Failure with preservation of secure state"	Yes	No conflict

FMT_LIM.1 "Limited capabilities"		No conflict.
FMT_LIM.2 "Limited availability"		No conflict.
FAU_SAS.1 "Audit storage"		No conflict.
FPT_PHP.3 "Resistance to physical attack"	Yes	No conflict
FDP_ITT.1 "Basic internal transfer protection"	Yes	No conflict.
FDP_IFC.1 "Subset information flow control"		No conflict.
FPT_ITT.1 "Basic internal TSF data transfer protection"	Yes	No conflict.
FCS_RND.1 "Quality metric for random numbers"		No conflict.
FPT_TST.2 "Subset TOE security testing"		No conflict
FDP_ACC.1 "Subset access control"		No conflict
FDP_ACF.1 "Security attribute based access control"		No conflict
FMT_MSA.3 "Static attribute initialisation"		No conflict
FMT_MSA.1 "Management of security attributes"		No conflict
FMT_SMF.1 "Specification of Management functions"		No conflict.
FCS_COP.1 (3DES) "Cryptographic operation"		No conflict
FCS_COP.1 (RSA) "Cryptographic operation"		No conflict
FCS_CKM.1 (RSA) "Cryptographic key generation"		No conflict
FDP_SDI.1 "Stored data integrity monitoring"		No conflict
FDP_SDI.2 "Stored data integrity monitoring and action"		No conflict

4. SECURITY OBJECTIVES

4.1 SECURITY OBJECTIVES FOR THE TARGET OF EVALUATION

The TOE shall achieve the following IT security objectives.

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

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 (EEPROM) thus prohibiting other Loaded-Applications from using it).

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

O.EFFECT_REMOVE
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 (Reference TEC-MAO-101-004/v4.3.1).

4.2 SECURITY OBJECTIVES FOR THE OPERATIONAL ENVIRONMENT

4.2.1 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.

4.2.2 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.

4.2.3 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 Personaliser 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).

4.2.4 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.

4.2.5 Objectives on Phase 7

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

4.2.6 Additional objectives for the operational environment

The following objectives are not ensured by the TOE (i.e. the Application Abstract Machine and the Application Memory Manager) but by the other components of the MULTOS OS.

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

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.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.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.

O.DECIPHER
The TOE shall provide a means to decipher the KTUs.

4.2.7 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 suitably protected (ie 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. EXTENDED COMPONENTS DEFINITION

None.

6. SECURITY REQUIREMENTS

This chapter describes the security functional requirements for the TOE and the security assurance requirements for the TOE.

6.1 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.

6.1.1 Security Audit Automatic Response (FAU_ARP)

6.1.1.1 FAU_ARP.1 Security alarms

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

6.1.2 Security audit analysis (FAU_SAA)

6.1.2.1 Potential violation analysis

FAU_SAA.1.1 Abend 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 Abend Iteration. The TSF shall enforce the following rules for monitoring audited events:

a) Accumulation or combination of:

- **An application attempts to execute MEL code outside of its code space or the code space of the codelet that it calls**
- **An application attempts to access data outside of its data space or the Public data segment**

known to indicate a potential security violation.
b) **none**.

6.1.3 Access control policy FDP_ACC

6.1.3.1 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.

6.1.4 Access control functions FDP_ACF

6.1.4.1 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, 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.
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.**

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.

6.1.5 Rollback (FDP_ROL)

6.1.5.1 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.**

6.1.6 Fail secure (FPT_FLS)

6.1.6.1 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 application attempts to execute MEL code outside of its code space or the code space of the codelet that it calls

b) An application attempts to access data outside of its data space or the Public data segment

6.1.7 Trusted recovery (FPT_RCV)

6.1.7.1 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 Control SF	Reset/power down during command processing
Application Delete Control SF	Reset/power down during command processing
Reset Protection SF	Reset/power down during command processing or application execution

6.1.8 Resource allocation (FRU_RSA)

6.1.8.1 FRU_RSA.1 Maximum quotas

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

6.2 SECURITY ASSURANCE REQUIREMENTS (SARS)

The security assurance requirement level is EAL7.

6.3 SECURITY FUNCTIONAL REQUIREMENTS DEPENDENCIES

This section demonstrates that all dependencies between components of security functional requirements included in this PP are satisfied.
Table 10 lists all functional components including security requirements in the IT environment. For each component, the dependencies specified in Common Criteria are listed, and an indication if these dependencies are satisfied for this TOE.

Security functional requirements	Dependencies in CC part 2	Dependencies in TOE
FAU_SAA.1: Potential Violation Analysis	FAU_GEN.1	No
FDP_ACC.2: Complete Access Control	FDP_ACF.1	FDP_ACF.1
FDP_ACF.1: security attributes based Access Control	FDP_ACC.1, FMT_MSA.3	FDP_ACC.2
FPT_FLS.1: failure with preservation of secure state	None	None
FPT_RCV.4: Function recovery	None	None
FAU_ARP.1: Security Alarms	FAU_SAA.1	FAU_SAA.1
FDP_ROL.1: Basic Rollback	FDP_ACC.1	FAU_ACC.2
FRU_RSA.1: Maximum quotas	none	none

Table 3: Functional dependencies in Multi-Application environment

Incompatible dependencies are explained as follows:

- Dependencies on FDP_ACC.1 are replaced by FDP_ACC.2 (a higher hierarchical component).
- 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.
- The dependencies of FDP_ACF.1 on FMT_MSA.3 are not applicable because the security attributes used in Load Application SFP have no management rule.

6.4 SECURITY REQUIREMENTS RATIONALE

6.4.1 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.

6.4.1.1 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.FLAW	O.ROLLBACK	O.RESOURCE	O.EFFECT_LOAD	O.EFECT_REMOVE	O.SEGREGATE
EAL7 requirements	X					
FAU_ARP.1			X			
FAU_SAA.1			X			
FDP_ACC.2						X
FDP_ACF.1				X	X	X
FDP.ROL.1		X				
FPT_FLS.1		X		X	X	X
FPT.RCV.4		X		X	X	
FRU_RSA.1			X			

Table 4: Mapping of security functional requirements and objectives

6.4.1.2 SFRs Justifications Rationale

This sub-section describes and justifies how the security objectives for the TOE are met by the security functional requirements.

O.FLAW*
The objective is met by formal design and high-quality testing as specified by EAL7 conformity requirements.

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 rollback of the application's code and data automatically.

O.RESOURCE
Resource preservation objective is assured by the following:
Maximum quotas: FRU_RSA.1 supported by FAU_ARP.1 (Security Alarms) and FAU_SAA.1 (Potential violation analysis).

O.EFFECT_LOAD
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.EFFECT_REMOVE
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).

6.4.2 SARs and the Security Requirements Rationale

The evaluation assurance level 7 is required to demonstrate that the TOE fulfills the most stringent Common Criteria requirements. In particular, the TOE design is formally described and its behavior is formally proved to correctly ensure its security objectives such as the segregation of Loaded-Applications (thanks to ADV requirements). Furthermore, the TOE is protected against sophisticated software and physical attacks (thanks to AVA_VAN.5 requirements).

6.4.2.1 ADV_SPM.1 Formal TOE security policy model

The formally modeled security policies consist of the segregation of the application executions, in particular:

1. If an application does not use the delegation mechanism, then its execution is contained in its memory space, i.e. this application cannot access to the data of the other applications (co-existing) on the TOE.
2. If an application uses the delegation mechanism, then its execution is contained in the memory space of the applications that have the delegation relation with it. In other words, this application cannot access to the data of another application that is not related to it by the delegation relation.

6.4.2.2 ADV_FSP.6 Complete semi-formal functional specification with additional formal specification

6.4.2.3 ADV_TDS.6 Complete semi-formal modular design with formal high-level design presentation

6.4.2.4 ADV_IMP.2 Implementation of the TSF

6.4.2.5 ADV_INT.3 Minimally complex internals

6.4.2.6 ATE_DPT.4. Testing: implementation representatio

6.4.2.7 ATE_COV.3. Rigorous analysis of coverage

6.4.2.8 ATE_FUN.2. Ordered Functional Testing

6.4.2.9 ALC_CMC.5 Advanced support

6.4.2.10 ALC_LCD.2 Measurable life-cycle model

6.4.2.11 ALC_TAT.3 Compliance with implementation standards – all parts

6.4.2.12 ALC_DVS.2 Sufficiency of security measures

6.4.2.13 AVA_VAN.5 Advanced methodical vulnerability analysis

7. TARGET OF EVALUATION SUMMARY SPECIFICATION

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

7.1 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 12 shows how these security functions satisfy the TOE security functional requirements.

7.1.1 Application Load Control SF (SF.Load)

SF.Load 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 reloading the application onto the same MCD is 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 loads of the application.

SF.Load ensures that 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 EEPROM, SF.Load checks if there is enough space available. If it is not the case, SF1 returns an error.

If load application fails, SF.Load ensures that the temporary loaded-application is erased on the next reset.

Permutational/probabilistic/cryptographic mechanisms used in this security function: None.

7.1.2 Application Delete Control SF (SF.Delete)

SF.Delete checks that the Application ID extracted from the Application Delete Certificate matches to a loaded application. Otherwise, no deletion is done.

Permutational/probabilistic/cryptographic mechanisms used in this security function: None.

7.1.3 Application Execution Management SF (SF.Firewall)

SF.Firewall 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.

SF.*Firewall* maintains separate storage and execution space for applications loaded onto an MCD. SF.*Firewall* manages a *pool of loaded applications. SF.Firewall 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 EEPROM 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 SF.Firewall. SF.Firewall 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. SF.*Firewall* 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 SF.*Firewall*. This ensures an application is unable to compromise the integrity or confidentiality of the code of other applications loaded on the MCD.

SF.*Firewall* ensures no application is able to write to the code space of any application, including itself. SF.*Firewall* 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 SF.*Firewall* and the application is terminated.

SF.Firewall 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 SF.Firewall. SF.*Firewall* 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 SF.Firewall. SF.Firewall 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 SF.Firewall writes all information related to the execution of the delegating application to an area under its control then begins the 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 the execution from the point of delegation.

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.

SF.Firewall 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 *SF.Firewall*. *SF.Firewall* 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 *SF.Firewall*. This ensures no application is able to compromise the integrity of MULTOS or the confidentiality of its sensitive information.

7.1.4 Start-up Initialization SF (SF.Init)

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

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

b) SF.Init 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).

c) SF.Init 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.

d) SF.Init rolls back any uncommitted writes in the Data Item Buffer.

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 SF.Init 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 SF.Init marks it as ACTIVE and when the source data item is successfully and completely updated SF.Init marks the data item copy as USED. If the card is reset and SF.Init found an ACTIVE block on the stack, SF.Init 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.

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.
EEPROM	Electronically Erasable Programmable Read Only Memory.
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	Organisational 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 non-volatile memory of 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).

External References

[1] Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model, Version 3.1, Revision 3, July 2009.

[2] Common Criteria for Information Technology Security Evaluation, Part 2: Security functional requirements, Version 3.1, Revision 3, July 2009.

[3] Common Criteria for Information Technology Security Evaluation, Part 3: Security assurance requirements, Version 3.1, Revision 3, July 2009.

[4] Common Criteria for Information Technology Security Evaluation, Protection Profile, Smartcard Integrated Circuit with Multi-Application Secure Platform, Version 2.0, November 2000, registered by French Certification Board under number PP/0010.

[5] BSI reports BSI-DSZ-CC-0640-2010 & BSI reports BSI-DSZ-CC-0606-2010 (reassessment 17 May 2011)

[6] Infineon Technologies SLE78CXxxxP and Derivates Security Target-lite (relating to the IC product certified by the BSI under BSI-DSZ-CC-0606-2010)

Other References

[HW-Manual]	Infineon Technologies *Hardware Reference Manual, 2010*
[ST_full_IDMotion]	ID Motion V1: (Public) Security Target.