TÜBİTAK BİLGEM

CENTER OF RESEARCH FOR ADVANCED TECHNOLOGIES
INFORMATICS AND INFORMATION SECURITY

AKİS PROJECT GROUP

UKİS (ULUSAL AKILLI KART İŞLETİM SİSTEMİ) V1.2.2 ON UKT23T64H V4

SECURITY TARGET

Revision No	21
Revision Date	03.09.2012
Document Code	UKIS-ST-Lite
File Name	UKIS_ST.DOC

Date of Revision

Revision No	Revision Reason	Date of Revision
01	Update from V1.2.1 to V1.2.2	11.11.2010
02	Update	19.04.2011
03	Update	04.05.2011
04	Update	12.07.2011
05	Update	15.07.2011
06	Update	03.08.2011
07	Update	16.08.2011
08	Update	19.09.2011
09	Update	17.11.2011
10	Update	22.11.2011
11	Update	14.12.2011
12	Update	06.02.2012
13	Update	23.02.2012
14	Update	08.03.2012
15	Update	04.05.2012
16	Update	08.05.2012
17	Update	14.05.2012
18	Update	05.06.2012
19	Update	11.06.2012
20	Update	07.08.2012
21	ST-Lite Version Prepared.	03.09.2010

CONTENT

Abbreviations & Glossary

UKİS	**Ulusal Akıllı Kart İşletim Sistemi**
ACE	**Advanced Crypto Engine**
APDU	**Application Protocol Data Unit**
CC	**Common Criteria**
DF	**Dedicated File**
EAL	**Evaluation Assurance Level**
EF	**Elemantary File**
ES	**Embedded Software**
HW	**Hardware**
MF	**Master File**
NVM	**Nonvolatile memory**
RAM	**Random Access Memory**
ROM	**Read Only Memory**
SFP	**Security Function Policy**
ST	**Security Target**
SOF	**Strength of Function**
TOE	**Target of Evaluation**
TPDU	**Transmission Protocol Data Unit**
TSF	**TOE Security Functions**
DAC	**Discretionary Access Control.**
TSC	**TSF Scope of Control**
TSP	**TOE Security Policy**
UART	**Universal Asynchronous Receiver Transmitter**

Basic Software: It is the part of ES in charge of the generic functions of the Smartcard IC such as Operating System, general routines and Interpreters.
Card Holder: End user of the card
Dedicated Software: It is defined as the part of ES provided to test the component and/or to manage specific functions of the component.
Embedded Software: It is defined as the software embedded in the Smartcard Integrated Circuit. The ES is the part of the non-volatile memories of the Smartcard IC.
Embedded software developer: Institution (or its agent) responsible for the smartcard embedded software development and the specification of pre-personalization requirements.
Initialization: It is the process to write specific information in the NVM during IC manufacturing and testing (smartcard product life cycle phase 3) as well as to execute security protection procedures by the IC manufacturer. The information could contain protection codes or cryptographic keys.
Integrated Circuit (IC): Electronic component(s) designed to perform processing and/or memory functions.
IC designer: Institution (or its agent) responsible for the IC development.
IC manufacturer: Institution (or its agent) responsible for the IC manufacturing, testing, and pre-personalization.
IC packaging manufacturer: Institution (or its agent) responsible for the IC packaging and testing**.**
OS: Operating System
Personalizer: Institution (or its agent) responsible for the smartcard personalization and final testing.

Personalization data: Specific information in the non volatile memory during personalization phase.

Security Information: Secret data, initialization data or control parameters for protection system.

Smartcard: A credit sized plastic card which has a non volatile memory and a processing unit embedded within it.

Smartcard Issuer: Institution (or its agent) responsible for the smartcard product delivery to the smartcard end-user.

Smartcard product manufacturer: Institution (or its agent) responsible for the smartcard product finishing process and testing.

NVM Special Area: An area that is located in EEPROM before filesystem area including initialization and personalization key, configuration bytes, addresses of error counters and flags.

Terminal: Standard or specially designed smart card reader.

Terminal Program: Program running on standard or specially designed smart card reader.

Smart Card/Terminal Application Programmer: Programmer responsible for developing smart card or terminal programs.

TSF Scope of Control: The set of interactions that can occur with or within a TOE and are subject to the rules of the TSP.

TOE Security Policy (TSP): A set of rules that regulate how assets are managed, protected and distributed within a TOE.

1. SECURITY TARGET INTRODUCTION

1.1 Security Target Reference

ST Title: UKİS (Ulusal Akıllı Kart İşletim Sistemi) version 1.2.2 on UKT23T64H v4 Security Target, rev 20, August 7, 2012.

This Security Target describes the TOE, intended IT environment, security objectives, security requirements, security functions and all necessary rationale.

1.2 TOE Reference

TOE Identification: UKİS (Ulusal Akıllı Kart İşletim Sistemi) version 1.2.2 on UKT23T64H v4.

1.3 TOE Overview

1.3.1 TOE Definition

The TOE is UKIS version 1.2.2 Smart Card Operating System together with underlaying TUBİTAK-UEKAE secure smart card IC (UKTÜM) UKT23T64H v4. The hardware UKTUM IC UKT23T64H V4 is certified according to CC EAL5+ (AVA_VAN.5).

In order to clarify the ST, the reference documents are the following:

- ISO7816 standard
- UKTÜM Security Micro Controller Data Book

1.3.2 TOE usage and security features for operational use

The TOE is a Contact Based Smart Card comprising a hardware platform and Smartcard Embedded Software serving as Operating System.

The interface of the TOE to the outside world is over a smart card reader or access device such as POS (Point of Sale) machine. PC (over the smart card reader) or access device transmits the commands to the smart card. Incoming commands are interpreted by TOE and the response is transmitted back to the access device or to the PC over smart card reader (Figure 1).

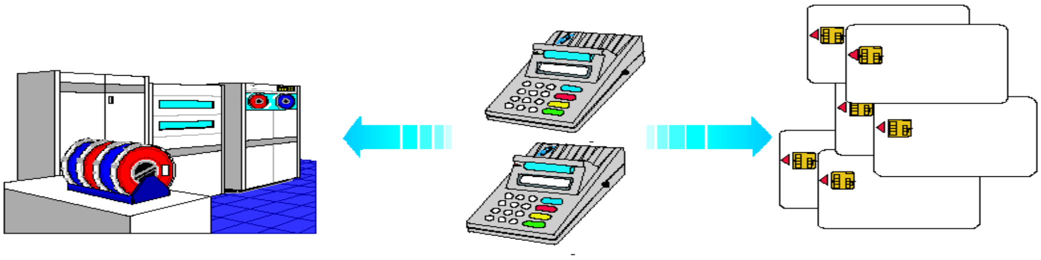


Figure 1. TOE's environment

The TOE has 8 pins according to the IEC/ISO 7816-2. TOE communicates with reader via I/O pin. VCC, GND, RST and CLK pins are used to operate the TOE. 2 pins are reserved for future use.

Basically TOE consists of 3 main parts:

- Metallic unit on plastic material which is called plastic module (physical plastic card)
- Silicon chip located in the metallic unit on the plastic module.
- Operating system (written in ROM and enables the operation of card functions using hardware units)

From the 3 parts listed above, HW and OS of the TOE are developed by TÜBİTAK-UEKAE. The first part is developed by a card manufacturer company (who provides the conditions that are presented in UKİS_TeslimveIsletim document) and the second part is developed by UEKAE YİTAL Department. TOE operates on UEKAE's secure IC UKT23T64H V4 chip. Embedded software specifications of the TOE are as follows:

- Embedded software (UKIS) of the TOE is loaded into ROM of the UEKAE's secure Smart Card chip UKT23T64H V4,
- Communicates with the PC via card reader according to ISO/IEC 7816-4 T = 1 protocol,
- Implements user and interface authentication,
- Manage the various kinds of data files stored in the non-volatile EEPROM memory
  - It is capable of binary file operations (open, update, erase, read),
  - Supports fixed length linear, variable length linear, fixed length cyclic file structures and file operations (open, append record, update record, read record),
  - Provides access control of the files if required, by the configuration
- Follows the life cycles (activation, manufacturing, initialization, personalization, administration, operation and death) and operates functions according to the present life cycle,
- Does not allow loading of executable files,
- Encrypts, decrypts, digitally signs and verifies with RSA/DES/3DES/AES cryptographic algorithms by using HW modules of the UKTÜM,
- Calculates SHA-1 hash.

As a smart card having the specifications above, the TOE can be used as PKI card (for digital sign), personal identification card and health care card.

1.3.3 Non-TOE hardware/software/firmware required by the TOE

There is no explicit non-TOE hardware, software or firmware required by the TOE to perform its claimed security features.

1.4 TOE Description

Algorithms and crypto specifications are;

Encryption;
DES-ECB: Plain data can be encrypted with a DES key.
DES3-ECB: Plain data can be encrypted with a DES3 key (A-B-A key structure).
AES-ECB: Plain data can be encrypted with an AES 256 bit key.
RSA: Plain data can be encrypted with an RSA 1024 or 2048 bits keys.

Decryption;
DES-ECB: Encrypted data can be decrypted with a DES key.
DES3-ECB: Encrypted data can be decrypted with a DES3 key (A-B-A key structure).
AES-ECB: Plain data can be decrypted with an AES 256 bit key.
RSA: Encrypted data can be decrypted with an RSA1024 or 2048 bits keys.

Digital Sign;
RSA: Plain data can be signed with an RSA 1024 or 2048 bits keys.

Digital Sign Verification;
RSA: Signed data with the length equal to the 1024 or 2048 bits length can be verified with an 1024 or 2048 bits length key.
Data Integrity;
DES-MAC: Cryptographic checksum is calculated with a DES-DES3 key.

Hash;
SHA-1: Data can be hashed with SHA-1 algorithm.

1.4.1 Physical Scope of TOE

The TOE comprises

- UKTÜM chip (the integrated circuit, IC): UKT23T64H V4
- The IC Embedded Software (UKİS v1.2.2 OS)
- UKiS User Manual.

1.4.1.1 UKTÜM UKT23T64H V4 Chip

The integrated circuit of the chip is UKTÜM UKT23T64H V4 (National Smart Card IC Version 4). UKT23T64H V4 comprises of 8051 CPU, 64K ROM , 64K NVM, 8K External RAM, UART, Timers, ACE (Advanced Crypto Engine), RNG (Random Number Generator), Security Sensors, IC dedicated library functions. UKTÜM UKT23T64H V4 has CC EAL 5+ (AVA_VAN.5) certificate. UKİS v1.2.2 Operating System is loaded into the ROM of the UKTÜM chip during the manufacturing of the IC.

The Integrated Circuit full description is available in the Security Target Document of UKT23T64H V4.

1.4.1.2 UKiS v1.2.2 Operating System

Operating System UKIS is embedded in ROM during chip manufacturing and can't be changed afterwards. However, data can be written into NVM under operating system's control.

1.4.1.2.1 UKiS v1.2.2 Operating System Components

Operating system components are

- Memory Manager
- File Manager
- Command Interpreter
- Communication Handler

Message is received by UART which is managed by communication handler in TOE. The message comes in TPDU (Transmission Protocol Data Unit) format. Incoming TPDU packet is analysed and block type decision is made by the communication handler. TPDU data is one of 3 different types of block, named R (Receive ready), S (Supervisor) and I (Information) block. R and S blocks are used to control the protocol. I block carries the command which is transmitted to the command interpreter and executed in TOE. When command execution is finished, communication handler sends the answer to the reader via UART. If the command is related with the file system, command interpreter calls the file manager. File manager is responsible for the operations in the file field which is in the NVM. Memory manager is used to open new file, close file, delete page and attach new page.

1.4.1.2.2 UKiS v1.2.2 Operating System Phases

A smart card life cycle process consists of some certain phases as shown in Figure 2. Just like the whole smart card, Operating System also consists of some phases which will be called as "OS Life cycle phases" (Figure 3) in order to obstruct confusion.

There are 7 different life cycle phases available on OS. Relations and crossing between these life cycle phases are shown in the Figure 3. Also there are some several keys available on TOE in order to be used within the execution of the secure commands. Command interpreter of TOE is designed to execute some special commands for the different life cycle phases.

These phases are;

Activation:

Main purposes of the activation life cycle phase is; check if the smart card includes correct TOE and load the initial values of the keys that will be used on the execution of the secure commands (initialization and personalization key).

Production:

Main purpose of the production phase is to format the NVM memory of the card and prepare the card for the next step. The next step would be the Initialization phase or the Administration phase depending on the format type. MF(Master File) is created in the Production phase.

Initialization:

Main purpose of the initialization phase is to load the initialization data into the card. Therefore the file system will begin to construct on the NVM on each command.

Personalization:

Main purpose of the personalization phase is to load the personalization data into the card. Henceforth the card will include unique data belonging to the end user.

Administration:

Administration phase is the management phase for the administrator and the authorized user. Changes in the file system or file system errors are handled in this phase. Smart cards with TOE have the reusability feature.

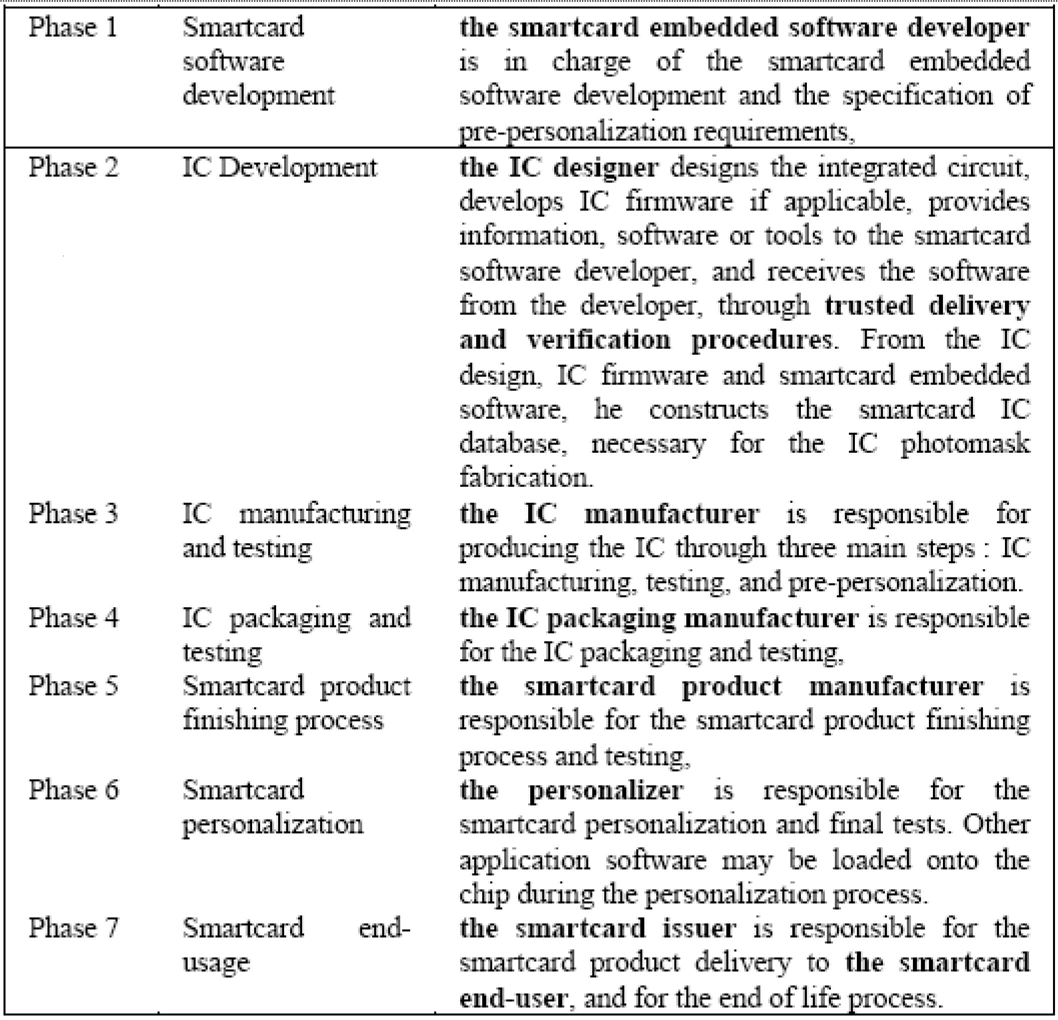
Operation:

In operation phase, TOE is also available for the end user.

Death:

When some security conditions are not satisfied or it is noticed that security is trying to be surpassed, TOE forces the card into death phase.

Smart cards and TOE have different life cycles, therefore two different life cycle schemas are shown in the following figures. The first one is life cycle of TOE which is shown in Figure 3 and the second one is Smart Card product life cycle which is shown in Figure 2.



Phase 1	Smartcard software development	**the smartcard embedded software developer** is in charge of the smartcard embedded software development and the specification of pre-personalization requirements,
Phase 2	IC Development	**the IC designer** designs the integrated circuit, develops IC firmware if applicable, provides information, software or tools to the smartcard software developer, and receives the software from the developer, through **trusted delivery and verification procedures**. From the IC design, IC firmware and smartcard embedded software, he constructs the smartcard IC database, necessary for the IC photomask fabrication.
Phase 3	IC manufacturing and testing	**the IC manufacturer** is responsible for producing the IC through three main steps : IC manufacturing, testing, and pre-personalization.
Phase 4	IC packaging and testing	**the IC packaging manufacturer** is responsible for the IC packaging and testing,
Phase 5	Smartcard product finishing process	**the smartcard product manufacturer** is responsible for the smartcard product finishing process and testing,
Phase 6	Smartcard personalization	**the personalizer** is responsible for the smartcard personalization and final tests. Other application software may be loaded onto the chip during the personalization process.
Phase 7	Smartcard end-usage	**the smartcard issuer** is responsible for the smartcard product delivery to **the smartcard end-user**, and for the end of life process.

Figure 2. Smart Card Product Life Cycle

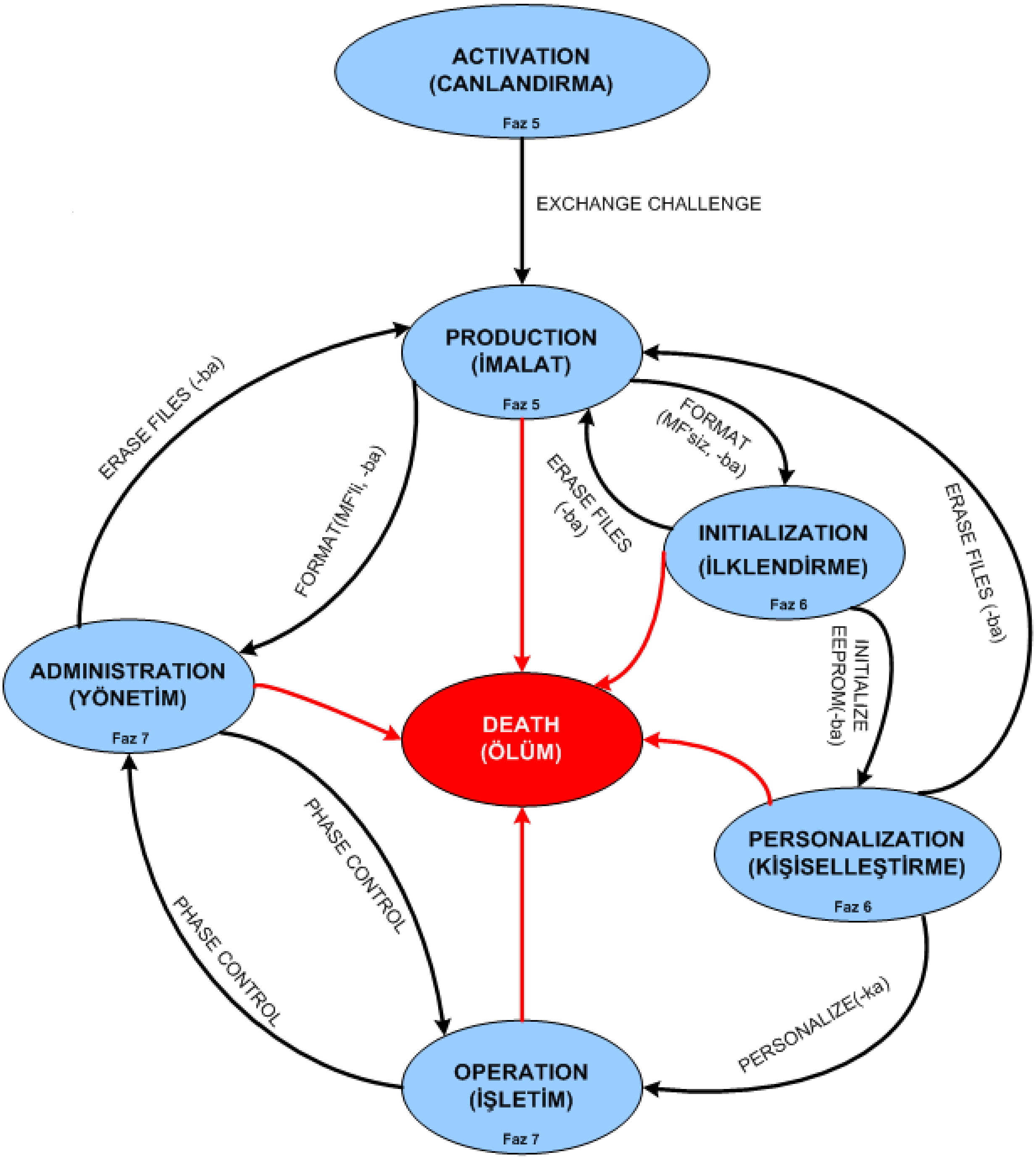


Figure 3. OS Life cycle phases

1.4.1.3 UKiS User Manual

Command set and operation of the UKiS v1.2.2 operating system is described in this document.

2 CONFORMANCE CLAIM

2.1 CC Conformance Claim

This ST claims conformance to

- Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model; CCMB-2006-09-001, Version 3.1, Revision 3, July 2009, [1]
- Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Components; CCMB-2007-09-002, Version 3.1, Revision 3, July 2009, [2], Extended
- Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Requirements; CCMB-2007-09-003, Version 3.1, Revision 3, July 2009, [3], Comformant

The

- Common Methodology for Information Technology Security Evaluation, Evaluation Methodology; CCMB-2007-09-004, Version 3.1, Revision 3, July 2009, [4]

has to be taken into account.

2.2 Protection Profile (PP) Claim

PP-9810 [5] and PP-0055[7] are used while writing this document, but ST has no PP claim.

2.3 Package Claim

Assurance Level: EAL 4 Augmented (AVA_VAN.5)

2.4 Conformance Rationale

An assurance requirement of EAL4 is required for this type of TOE since it is intended to defend against sophisticated attacks. In order to provide a meaningful level of assurance that the TOE provides an adequate level of defense against such attacks, the evaluators should have access to the low level design and source code. The lowest for which such access is required is EAL4. The assurance level EAL4 is achievable, since it requires no specialist techniques on the part of the developer.

EAL4 is augmented with AVA_VAN.5.

3 SECURITY PROBLEM DEFINITION

3.1 Introduction

This section includes the following:

- Threats
- Organizational security policies
- Assumptions

This information provides the basis for the Security Objectives specified in Section 4, the security functional requirements for the TOE in Section 5, and the TOE Security Assurance Requirements specified in Section 6.

The assets to be protected are:

- TOE system design,
- the basic software (including operating system programs and documentation),
- TOE hex code,
- the TSF data of the TOE (initialization and personalization requirements such as keys, PIN/PUK, filesystem).

These assets have to be protected in terms of confidentiality and integrity.

3.2 Threats

The TOE as defined in Chapter 2 is required to counter the threats described hereafter; a threat agent wishes to abuse the assets either by functional attacks or by environmental manipulation, by specific hardware manipulation, by any other type of attacks.

Threats have to be split into:

- threats which can be countered by the TOE (class I)
- threats which can be countered by the TOE environment (class II)

3.2.1 Threats on all smartcard product life cycle phases (1 to 7), Figure 2

The threat agents are very general and are case dependent.

- During phase 1 to 3, developers are the most apt to mount the threats.
- During phase 1 to 3, external persons can spy on communications or steal the TOE so to attack it. They have fewer capabilities than the developers, but they cannot be screened out.
- For phases 4 to 6, the main potential threat agents are personnel allowed manipulating the TOE or personalization data, but external parties can also be active.
- During phase 7, the administrator, issuer, or at least its agents, can in some cases be considered a threat agent.
- During phase 7, in some cases, such as electronic purses, the card holder can be interested in breaking the TOE.
- During phases 3 to 7, threats coming from outsiders must be preceded by the stealing of the TOE.

T.CLON Functional cloning of the TOE (full or partial) appears to be relevant to any phase of the smart card product life-cycle, from phase 1 to phase 7.
Generally, this threat is derived from specific threats combining unauthorized disclosure, modification or theft of assets at different phases.
Assets that are subject to threats:

Phase 1: TOE system design, basic operating system and documentation
Phase 2: Basic operating system, TOE hex code
Phase 3: TOE hex code
Phase 4: TOE hex code
Phase 5: TOE hex code
Phase 6: TSF data
Phase 7: TSF data

T.DIS Unauthorized disclosure of the smartcard embedded software, data or any related information.
Assets that are subject to threats:
Phase 1: TOE system design, basic operating system and documentation
Phase 2: Basic operating system, TOE hex code
Phase 3: TOE hex code
Phase 4: TOE hex code
Phase 5: TOE hex code
Phase 6: TSF data
Phase 7: TSF data

T.MOD Unauthorized modification of the smartcard embedded software and data.
Assets that are subject to threats:
Phase 1: TOE system design, basic operating system and documentation
Phase 2: Basic operating system, TOE hex code
Phase 3: TOE hex code
Phase 4: TOE hex code
Phase 5: TOE hex code
Phase 6: TSF data
Phase 7: TSF data

3.2.2 Threats on smartcard product life cycle phase 1

During phase 1, two types of threats have to be considered:

a) Threats on the smartcard embedded software and its development environment,
b) Threats on software development tools coming from the IC manufacturer.

The main threat agents are developers, but they can also be other parties working in the same company or outside.

T.T_TOOLS Theft or unauthorized use of the smartcard embedded software development tools (such as PC, databases, etc). TOE system design, basic software, TOE hex code are subject to threats.

T.FLAW Introduction of flaws in the embedded system due to malicious intents or insufficient development. TOE system design, basic software, TOE hex code are subject to threats.

T.T_SAMPLE Theft or unauthorized use of integrated circuit samples containing the embedded software (e. g. bound out, dil, etc). TOE hexcode is subject to threat.

T.MOD_INFO Unauthorized modification of any information (technical or detailed specifications, implementation code, design technology, tools characteristics) used for developing software or loading data. TOE system design, basic software, TOE hex code are subject to threats.

T.DIS_TEST Unauthorized disclosure of the smartcard embedded software test information including interpretations. TSF data is subject to threat.

T.DIS_INFO Unauthorized disclosure of any information (technical or detailed specifications, implementation code, design technology, tools characteristics) used for developing software or loading data. This includes sensitive information on IC specification, design and technology, software and tools. TOE system design, basic software, TOE hex code are subject to threats.

3.2.3 Threats on delivery of software and related information from smartcard product life cycle phases 1 and 2 to smartcard product life cycle phases 2, 3 and 6

These threats address

- software to be embedded send by the software developer to the IC designer (for designing the photomask): phase 1 to phase 2,
- Transformed software send from the IC designer to the IC manufacturer: phase 2 to phase 3,
- prepersonalization data send by software developer to IC manufacturer for prepersonalization, phase 3: phase 1 to phase 3,
- personalization data send by software developer to the personalizer, phase 1 to phase 6.

Data send directly from smartcard issuer to the IC manufacturer and to personalizer are considered as belonging respectively to phase 3 and phase 6.
The main threats agents are eavesdroppers on networks or on other delivery processes.

T.T_DEL Theft or unauthorized use of the smartcard embedded software and any additional TSF data delivered to the IC designer, IC manufacturer or to the personalizer. TOE hex code and TSF data are subject to threats.

T.MOD_DEL Unauthorized modification of the smartcard embedded software and any additional TSF data delivered to the IC designer, IC manufacturer or to the personalizer. TOE hex code and TSF data are subject to threats.

T.DIS_DEL Unauthorized disclosure of the smartcard embedded software and any additional TSF data delivered to the IC designer, IC manufacturer or to the personalizer. TOE hex code and TSF data are subject to threats.

3.2.4 Threats on smartcard product life cycle phase 2

The main threat agents are persons working inside the IC designing plant or persons breaking in.

T.DIS_TEST Unauthorized disclosure of the smartcard embedded software test information including interpretations. TOE hex code is subject to threat.

3.2.5 Threats on smartcard product life cycle phases 3 to 6

The main threats agents are persons working inside the plants or working for the agents responsible for transportation between plants.

T.T_PRODUCT Theft or unauthorized use of the smartcard product or any related information. For example, unauthorized use of the embedded software application functions. TOE hex code and TSF data are subject to threats.

T.DIS_TEST Unauthorized disclosure of the smartcard embedded software test information including interpretations. TOE hex code and TSF data are subject to threats.

3.2.6 Threat on smartcard product life cycle phases 3 to 7 (included)

The main threats agents are persons working inside the plants or working for the agents responsible for transportation between plants or from outside parties who first steal the smartcard product.

T.INFO_LEAKAGE: Exploit of information which is leaked from the TOE during its usage in order to disclose confidential TSF data. The information leakage may be inherent in the normal operation or caused by the attacker. Leakage may occur through emanations, variations in power consumption, I/O characteristics, clock frequency, or by changes in processing time requirements. Examples are the Differential Electromagnetic Analysis (DEMA) and the Differential Power Analysis (DPA). Moreover the attacker may try actively to enforce information leakage by fault injection (Differential Fault Analysis). TSF data and TOE hex code are subject to threat.

T.PHYS_TAMPER: An attacker may perform physical probing of the chip in order to disclose TSF Data or to disclose/reconstruct the Embedded Software.

An attacker may physically modify the chip in order to

(i) modify security features or functions of the chip,
(ii) modify security functions of the Embedded Software,
(iii) to modify TSF data.

The physical tampering may be focused directly on the disclosure or manipulation of TOE TSF Data (keys) or indirectly by preparation of the TOE to following attack methods by modification of security features (to enable information leakage through power analysis). Physical tampering requires direct interaction with the chip internals. Techniques commonly employed in IC failure analysis and IC reverse engineering efforts may be used. Before that, the hardware security mechanisms and layout characteristics need to be identified.

Determination of software design including treatment of TSF Data may also be a pre-requisite. The modification may result in the deactivation of a security function. Changes of circuitry or data can be permanent or temporary.

TSF data and TOE hex code are subject to threat.

3.2.7 Threat on smartcard product life cycle phase 7

The threat can come from outside parties who first steal the smartcard product. The realisation of the threat is a first step toward breaking open the product.

T.T_PRODUCT Theft or unauthorized use of the smartcard product or any related information. For example, unauthorized use of the embedded software application functions. TOE hex code and TSF data are subject to threat.

The table given below indicates the relationship between the smartcard life-cycle phases, the threats and the type of the threats.

Threats	Phase 1	Phase 2	Phase 3	Phase 4	Phase 5	Phase 6	Phase 7
Functional Cloning							
T.CLON	Class II	Class II	Class I/II	Class I/II	Class I/II	Class I/II	Class I/II
Unauthorized disclosure of assets							
T.DIS	Class II	Class II	Class I/II	Class I	Class I	Class I/II	Class I
T.DIS_INFO	Class II						
T.DIS_DEL	Class II	Class II	Class II			Class II	
T.DIS_TEST	Class II	Class II	Class II	Class II	Class II	Class II	
T.INFO_LEAKAGE			Class I	Class I	Class I	Class I	Class I
T_PHYS_TAMPER			Class I	Class I	Class I	Class I	Class I
Theft of assets							
T.T_TOOLS	Class II						
T.T_SAMPLE	Class II						
T.T_DEL	Class II	Class II	Class II			Class II	
T.T_PRODUCT			Class I/II	Class I/II	Class I/II	Class I/II	Class I/II
Unauthorized modification or faulty development of assets							
T.FLAW	Class II						
T.MOD	Class II	Class II	Class I/II	Class I	Class I	Class I/II	Class I
T.MOD_INFO	Class II						
T.MOD_DEL	Class II	Class II	Class II			Class II	

Table 1 – Threats during phases

3.3 Organizational Security Policies

No security policy has been defined within the scope of this ST.

3.4 Assumptions

This section concerns assumptions about;

1. Security aspects of the environment in which the TOE is intended to be used;
   - assumptions on the TOE delivery process from phase to phase,
   - assumptions on IC development,
   - assumptions on smartcard product life cycle phases 2 to 7.
2. The intended usage of the TOE.

3.4.1.1 Assumptions on the TOE delivery process from phase to phase

A.DLV_CONTROL procedures must guarantee the control of the TOE delivery and storage process and conformance to its objectives as described in the following secure usage assumptions. Secure storage and handling procedures are applicable for all TOE's parts (programs, data, documents, etc).

A.DLV_CONF procedures must also prevent if applicable any non-conformance to the confidentiality convention and must have a corrective action system in case any non-conformance or misprocessed procedures are identified.

A.DLV_PROTECT procedures shall ensure protection of material/information under delivery including the following objectives:

- non-disclosure of any security relevant information,
- identification of the elements under delivery,
- meeting confidentiality rules (confidentiality level, transmittal form, reception acknowledgment), physical protection to prevent external damage.

A.DLV_TRANS procedures shall ensure that material/information is delivered to the correct party.

A.DLV_TRACE procedures shall ensure traceability of delivery including the following parameters:

- origin and shipment details,
- reception, reception acknowledgment,
- location material/information.

A.DLV_AUDIT procedures shall ensure that corrective actions are taken in case of improper operation in the delivery process and highlight all non-conformances to this process.

A.DLV_RESP procedures shall ensure that people dealing with the procedures for delivery have got the required skill, training and knowledge to meet the procedure requirements and to act to be fully in accordance with the above expectations.

3.4.1.2 Assumptions on IC development (smartcard product life cycle phase 2)

There is one type of assumptions: the assumptions on the personnel aspects.

Secure personnel assumptions:

A.IC_ORG procedures dealing with physical, personnel, organizational, technical measures for the confidentiality and integrity of smartcard embedded software and data (e.g. source code and any associated documents) shall exist and be applied in the smartcard IC database construction.

3.4.1.3 Assumptions on smartcard product life cycle phases 3 to 6

A.USE_TEST it is assumed that appropriate functionality testing of the smartcard functions is used in phases 3 to 6.
A.USE_PROD it is assumed that security procedures are used during all manufacturing and test operations through smartcard production phases to maintain the confidentiality and integrity of the TOE and of its manufacturing and test data (to prevent any possible copy, modification, retention, theft or unauthorized use).

3.4.1.4 Assumption on smartcard product life cycle phase 7

A.USE_SYS it is assumed that the security of sensitive data stored/handled by the system (terminals, communications ...) is maintained.

3.4.1.5 Assumption on the intended usage of the TOE, related with TOE Life Cycle Phases (Figure 3) except Activation phase

A.USE_OPR after giving a warning message from TSF for corrupted objects (DF-EF-DF PIN-DF PUK, System PIN, System PUK), it is assumed that the user and smart card/terminal application programmer know which corrupted objects can be used or not without taking any risk for security and availability of the TOE.

4 SECURITY OBJECTIVES

The security objectives of the TOE and its environment cover principally the following aspects:

- integrity and confidentiality of assets,
- protection of the TOE and associated documentation during development and production phases.

4.1 Security objectives for the TOE

The TOE shall use state of art technology to achieve the following TOE security objectives.

O.INTEGRITY The TOE must provide the means of detecting loss of integrity affecting security information stored in memories.
O.FUNCTION The TOE must provide protection against unauthorized use of its software application functions.
O.CLON The TOE functionality needs to be protected from cloning.
O.OPERATE The TOE must ensure the continued correct operation of its security functions.
O.DIS_MECHANISM The TOE shall ensure that the software security mechanisms are protected against unauthorized disclosure.
O.DIS_MEMORY The TOE shall ensure that the embedded software does not allow unauthorized access to information stored in memories.
O.MOD_MEMORY The TOE shall ensure that the embedded software does not allow unauthorized modification or corruption of the information stored in memories.
O.FLAW The TOE must not contain flaws in design, data values or implementation.
O.PROT_INF_LEAK The TOE must provide protection against disclosure of the Embedded Software (OS) and confidential TSF data stored and/or processed in the chip

- by measurement and analysis of the shape and amplitude of signals or the time between events found by measuring signals on the electromagnetic field, power consumption, clock, or I/O lines and
- by forcing a malfunction of the TOE and/or
- by a physical manipulation of the TOE.

This objective pertains to measurements with subsequent complex signal processing due to normal operation of the TOE or operations enforced by an attacker. Details correspond to an analysis of attack scenarios which is not given here. PP-0055 [7] is referenced for this objective.

O.PROT_PHYS-TAMPER The TOE must provide protection of the confidentiality and integrity of the TSF Data, and the Embedded Software (OS). This includes protection against attacks with enhanced-basic attack potential by means of

- measuring through galvanic contacts or
- manipulation of the hardware and its security features, as well as controlled manipulation of memory contents (TSF Data) with a prior reverse-engineering to understand the design and its properties and functions.

PP-0055 [7] is referenced for this objective.

4.2 Security objectives for the environment

4.2.1 Objectives on smartcard product life cycle phase 1 (development phase)

O.SOFT_ACS The embedded software shall be accessible only by authorized personnel (physical, personnel, organizational, and technical procedures).

O.MECH_ACS Details of software security mechanisms shall be accessible only by authorized personnel.
O.TI_ACS Security relevant technology information shall be accessible only by authorized personnel. This information includes software test information including the interpretations of the test results.
O.INIT_ACS TSF data shall be accessible only by authorized personnel (physical, personnel, organizational, and technical procedures).
O.TOOLS_ACS Embedded software development tools shall be accessible only by authorized personnel.
O.SAMPLE_ACS Samples used to run test shall be accessible only by authorized personnel.

4.2.2 Objectives on phases smartcard product life cycle 2, 3 and 6 and on delivery to these phases.

O.DIS_DEV The IC designer and the personalizer must have procedures to control the sales, distribution, storage and usage of the software and classified documentation, suitable to maintain the integrity and the confidentiality of the assets of the TOE.
It must be ensured that tools are only delivered to the parties' authorized personnel.
It must be ensured that confidential information on defined assets is only delivered to the parties' authorized personnel.
O.SOFT_DLV The embedded software must be delivered from the smartcard software developer 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. The same goes for the delivery of the personalization data from the product manufacturer to the personalizer.

4.2.3 Objectives on smartcard product life cycle phase 2 to 7

O.PRODUCT_DEV The IC designer, manufacturer, personalizer and issuer must have procedures to control the sales, distribution, storage and usage of the product, suitable to maintain the integrity and the confidentiality of the assets of the TOE. This applies also to test information whenever it is pertinent.
It must be ensured that the product is only delivered to the parties' authorized personnel and authorized end users.

4.2.4 Objective on the intended usage of the TOE, related with TOE Life Cycle Phases (Ref : ST Figure 3) except Activation phase

O.OPERATION TOE users must take training periodically on using the corrupted objects. (DF, EF, DF PIN, DF PUK, System PIN and System PUK). User and administrator guide defines how the user or administrator shall act upon detection of any corruption on DF, EF, DF PIN, DF PUK, System PIN and System PUK.

4.3 Security objectives rationale

This section demonstrates that the stated security objectives counter all the identified threats and consistent with the identified assumptions.

4.3.1 Security Objectives Related with Threats

The following tables show which security objectives counter which threats phase by phase. It demonstrates that at least one security objective is correlated to at least one threat, and that each threat is countered by at least one objective.

4.3.1.1 Classes of threats relative to smart card product life cycle phases

As shown in Table 1, threats can be expected in different phases of the TOE life-cycle, and can be countered either by the TOE (class I) or by the environment (class II) or by both. The TOE is designed during phase 1, but is constructed only at the end of phase 3.

T.CLON Cloning can be done at any phase of card life. During phases 1 and 2, as the product is not materialized, it cannot contribute to countering the threat. During these phases, threat T.CLON can only be met by security objectives for the environment. TOE samples are finished products which are used during phase 1 for evaluations, and can help to counter T.CLON, but still the security objectives for the environment must be sufficient to meet the threat. For the remaining phases, 3 to 7, the TOE participates to countering the threats, but environment security procedures must still be applied.

T.DIS Disclosure of software and data can be done at any phase of card life. During phases 1 and 2, as the product is not materialized, it cannot contribute to countering the threat and then only environmental procedures counter the threat. For the remaining phases, 3 to 7, the TOE counters the threats on embedded software and data. During the phases 3 and 6, more data are loaded in the TOE, so environmental procedures must also be taken to counter the threat.

T.DIS_INFO The threat concerns data used for developing software. This data is present only during Smartcard software development, phase 1.

T.DIS_DEL This threat is relative to delivery of information, software and/or data from phase 1 (software developer) to phase 2 (IC designer) and phase 3 (IC manufacturer). Part of the data, software, is transferred in a modified form from phase 2 to phase 3. Delivery to personalizer (phase 6), can come from the software developer (phase 1) or from the smartcard issuers, in which case it is considered inside the phase 6. As the data is not yet implemented in the TOE, the threat can only be countered by environment procedures.

T.DIS_TEST Tests are conducted at the end of phases 1, 2, 3, 4, 5, 6. These tests being part of the environmental procedures, this threat is countered by environmental procedures.

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

T.T_SAMPLE TOE samples are used only during phase 1, therefore this threat only exists during phase 1. The theft or unofficial use of samples is countered by environmental procedures.

T.T_DEL This threat is relative to delivery of information, software and/or data from phase 1 (software developer) to phase 2 (IC designer) and phase 3 (IC manufacturer). Part of the data, software, is transferred in a modified form from phase 2 to phase 3. Delivery to personalizer (phase 6), can come from the software developer (phase 1) or from the smartcard issuers, in which case it is considered inside the phase 6. As the data is not yet implemented in the TOE, the threat can only be countered by environment procedures.

T.T_PRODUCT The product exists only from phase 3 on. The threat can only be carried out during phases 3 to 7. The threat is partly met by environmental procedures. The product, when manufactured (phases 3 to 7) also counters the threat by limiting usage to the authenticated rightful owners.

T.FLAW Flaws in the design of the TOE can only be introduced during the development phase (phase 1).

T.MOD Modification of software and data can be done at any phase of Smartcard life cycle. During phases 1 and 2, as the product is not materialized, it cannot contribute to counter the threat. During the beginning of phase 3, (test phase) the TOE cannot counter the threat, but at the end (once the fuse has been blown), the TOE participates to countering it. For the remaining phases, 3 to 7, the TOE counters the threats on embedded software and data. During

personalization phase (phase 6) more data is loaded, so that environmental procedures must also be taken to counter the treat.
T.MOD_INFO The threatened information is only used for software development, so it can only be modified during phase 1.
T.MOD_DEL This threat is relative to delivery of information, software and/or data from phase 1 (software development) to phase 2 (IC designer) and phase 3 (IC manufacturer). Part of the data, software is transferred in a modified form, from phase 2 to phase 3. Delivery to personalizer (phase 6), can come from the software developer (phase 1) or from the smartcard issuers, in which case it is considered inside the phase 6.. As the data is not yet implemented in the TOE, the threat can only be countered by environment procedures.
T.INFO_LEAKAGE This threat concerns phases 3 to 7 including 7.
T.PHYS_TAMPER This threat concerns phases 3 to 7 including 7.

4.3.1.2 Threats addressed by security objectives for the TOE

The product is constructed only after the end of smart card product life cycle phase 3 , therefore it can only meet functional requirements during smart card product life cycle phases 3 to 7. The threats to be addressed by the TOE are: T.CLON, T.DIS, T.T_PRODUCT, T.MOD, T.INFO_LEAKAGE, and T.PHYS_TAMPER.
The threat T.FLAW which appears only in phase 1 is to be covered by the TOE development methodology.

O.INTEGRITY addresses the integrity of the TOE once it is completed, thus it counters the threat T.MOD and T_PHYS_TAMPER during smart card product life cycle phases 3 to 7.
O.FUNCTION addresses illegal use of the TOE, thus it counters the threat T.T_PRODUCT during smart card product life cycle phases 3 to 7. It also counters the use of a duplicate of the TOE, thus it counters T.CLON.
O.OPERATE Correct operations of the TOE security functions assures that its confidential information cannot be disclosed, threat T.DIS, T_INFO_LEAKAGE and that the operations cannot be corrupted, T.MOD, T_PHYS_TAMPER during smart card product life cycle phases 3 to 7.
O.FLAW addresses the threat T.FLAW during the conception of the TOE. This objective allows the TOE to counter the threats T.DIS, T_INFO_LEAKAGE, T.MOD and T_PHYS_TAMPER once it is manufactured, smart card product life cycle phases 3 to 7.
O.DIS_MECHANISM addresses the threats T.DIS and T_PHYS_TAMPER. It helps to counter T.MOD and T_INFO_LEAKAGE by keeping confidential the security mechanisms which have to be broken to realize the threat. As knowledge of the security mechanisms is necessary for cloning, it also contributes to counter T.CLON. The TOE can fulfill this objective during smart card product life cycle phases 3 to 7.
O.DIS_MEMORY addresses the disclosure of TOE memory, threats T.DIS and T_INFO_LEAKAGE . As knowledge of memory content is necessary for cloning, T.CLON is also addressed. The TOE can fulfill this objective during smart card product life cycle phases 3 to 7.
O.MOD_MEMORY addresses the modification of TOE memory, threat T.MOD and T_PHYS_TAMPER. The TOE can fulfill this objective during smart card product life cycle phases 3 to 7.
O.CLON addresses the cloning of the TOE, threat T.CLON. By extension, this objective addresses the unauthorized use of embedded software functions which is part of T.T_PRODUCT. The TOE can fulfill this objective during smart card product life cycle phases 3 to 7.

O.PROT_INF_LEAK addresses the disclosure of data, threats T_DIS, T.INFO_LEAKAGE and T.CLON. The TOE can fulfill this objective during smart card product life cycle phases 3 to 7.

O.PROT_PHYS_TAMPER addresses the confidentiality and integrity of the TOE, threats T_MOD, T.PHYS_TAMPER and T.CLON. The TOE can fulfill this objective during smart card product life cycle phases 3 to 7.

Threats/T.Obj.	O_INTEGRITY	O_FUNCTION	O_OPERATE	O_FLAW	O_DIS_ MECHANISM	O_DIS_ MEMORY	O_MOD_ MEMORY	O_CLON	O_PROT_INF_ LEAKAGE	O_PROT_PHYS _ TAMPER
T_CLON		X			X	X		X	X	X
T_DIS			X	X	X	X			X	
T_PRODUCT		X						X		
T_MOD	X		X	X	X		X			X
T_FLAW				X						
T_INFO_LEAKAGE			X	X	X	X			X	
T_PHYS_TAMPER	X		X	X	X		X			X

Table 2. Mapping of TOE objectives to threat

It is demonstrated that all class I threats and T.FLAW are addressed by at least one Security Objectives for the TOE.

4.3.1.3 Threats addressed by Security Objectives for the environment

4.3.1.3.1 Smart card product life cycle phase 1 enviromental Security Objectives

The threats present during phase 1 and which are not linked to delivery are:

- Threats occurring all the phases: T.CLON, T.DIS, and T.MOD

Threats occurring phases 3 to 7: T_INFO_LEAKAGE T_PHYS_TAMPER

- Threats specific to phase 1: T.DIS_INFO, T.DIS_TEST, T.T_TOOLS, T.T_SAMPLE, T.MOD_INFO

Threat T.FLAW is already addressed by the TOE development objective O.FLAW.

Threats T.T_DEL, T.MOD_DEL and T.DIS_DEL are considered later.

O.SOFT_ACS Restricting software access to authorized developers meets the threats T.DIS and T.MOD which require access to the software or related data. This knowledge is also necessary to mount threat T.CLON andT_PHYS_TAMPER ,T_INFO_LEAKAGE in later phases.

O.MECH_ACS Restricting access to the security mechanisms to authorized developers meets the threats T.DIS, T.MOD which require access to the software or related data. This knowledge is also necessary to mount threat , T_PHYS_TAMPER ,T_INFO_LEAKAGE and so T.CLON.

O.TI_ACS addresses disclosure and modification of related information. It thus addresses threats related to the illegal disclosure these information, T.DIS_INFO, and T.DIS_TEST or to their illegal modification T.MOD_INFO. This objective also helps addressing T.CLON, T.INFO_LEAKAGE and T. PHYS_TAMPER, threats easier to mount if related information is known.

O.INIT_ACS addresses the part of T.DIS and T.MOD concerning initialization information. As this information is necessary to construct a TOE, O.INIT_ACS also addresses T.CLON.

O.TOOLS_ACS addresses specifically the threat T_TOOLS. If complete knowledge of the embedded software is not known, the development tools are necessary to build replica of the TOE. Thus O.TOOLS_ACS addresses T.CLON.

O.SAMPLE_ACS addresses specifically T.T_SAMPLE. Possession of samples is also a great help to finding the embedded software and data so to clone the TOE. Thus O.SAMPLE_ACS addresses also T.DIS and T.CLON. Samples also help discovery of leakage behavior of the TOE, it helps T_INFO_LEAKAGE.

Threats/E.Obj	O_SOFT_ACS	O_MECH_ACS	O_TI_ACS	O_INIT_ACS	O_TOOLS_ACS	O_SAMPLE_ACS
T_CLON	X	X	X	X	X	X
T_DIS	X	X		X		X
T_MOD	X	X		X		
T_DIS_INFO			X			
T_DIS_TEST			X			
T_T_TOOLS					X	
T_T_SAMPLE						X
T_MOD_INFO			X			
T_INFO_ LEAKAGE	X	X	X			X
T_PHYS_TAMPER	X	X	X			

Table 3. Mapping of security objectives for the environment to threats relative to phase 1

It is demonstrated that all class II threats during phase 1 are addressed by at least one Security Objectives for the environment.

4.3.1.3.2 Smart card product life cycle phases 2 and delivery to smart card product life cycle phases 2, 3 and 6 enviromental Security Objectives.

These phases concern more specifically the IC designer, the IC developer and the Personalizer who have to load data into the TOE and must exchange data with the preceding phases. Delivery of TOE itself is not addressed here.

The threats to be addressed are:

- Threats occurring during all the phases: T.CLON, T.DIS, T.MOD
- Threats on phase 2: T.DIS_TEST
- Threats on delivery of data to phases 2, 3 and 6: T.T_DEL, T.MOD_DEL, T.DIS_DEL.

O.DIS_DEV During phase 2 software test information is used by IC designer and IC manufacturer.

O.DIS_DEV addresses threat T.DIS_TEST. Software data and personalization data is manipulated also during these phases so that O.DIS_DEV addresses also T.DIS and T.MOD. As the realization of these threats can lead to cloning, O.DIS_DEV addresses also T.CLON.

O.SOFT_DLV addresses specifically threats linked to delivery processes of data, T.T_DEL, T.MOD_DEL and T.DIS_DEL. As the realization of these threats allows T.CLON, T.DIS and T.MOD to be materialized, these threats are also addressed.

Threats/ E.Obj	DIS_DEV	SOFT_DLV
T_CLON	X	X
T_DIS	X	X
T_DIS_DEL		X
T_DIS_TEST	X	
T_DEL		X

T_MOD	X	X
T_MOD_DEL		X
T_PHYS_TAMPER	X	X
T_INFO_ LEAKAGE	X	X

Table 4. Mapping of security objectives for the environment to threats relative to phases 2 and to delivery to phases 2, 3 and 6

It is demonstrated that all class II threats during phases 2 and threats concerning delivery to phases 2, 3 and 6 are addressed by at least one security objectives for the environment.

4.3.1.3.3 Smart card product life cycle phases 3 to 7

The threats considered are those concerning the delivery of the product and it's management as well as threats using the physical characteristic of the IC in which the software and the data is embedded.
The threats are: T.CLON, T.DIS, T.MOD, T.T_PRODUCT, T_INFO_LEAKAGE, T_PHYS_TAMPER and T.DIS_TEST

O.PRODUCT_DEV contributes to the protection of TOE data and related information including test information during phases 3 and 6, and thus addresses T.DIS, T.MOD T.DIS_TEST. O.PRODUCT_DEV addresses directly T.T_PRODUCT and thus helps to counter T.CLON.

Threats/ E.Obj	PRODUCT_DEV
T_CLON	X
T_DIS	X
T_PRODUCT	X
T_DIS_TEST	X
MOD	X

Table 5. Mapping of security objectives for the environment to threats relative to phases 3 to 7

It is demonstrated that all class II threats during phases 3 to 7 are addressed by at least one Security Objectives for the environment.

4.3.2 Security Objectives Related with Assumptions

4.3.2.1 Security assumptions met by the Security Objectives for the environment (after the development phase)

This section demonstrates that the security assumptions are suitably satisfied by the identified security objectives for the environment.
Each of the security objectives for the environment is addressed by assumptions.
The following tables demonstrate which assumptions contribute to the satisfaction of each security objective. For clarity, the table does not identify indirect dependencies.

O.DIS_DEV is linked to A.DLV_CONTROL, A.DLV_CONF, A.DLV_PROTECT, A.DLV_TRANS, A.DLV_TRACE, A.DLV_AUDIT, A.DLV_RESP, A.IC_ORG, A.USE_PROD and A.USE_SYS
O.SOFT_DLV is linked to A.DLV_CONTROL, A.DLV_CONF, A.DLV_PROTECT, A.DLV_TRANS, A.DLV_TRACE, A.DLV_AUDIT, A.DLV_RESP, A.USE_TEST, A.USE_PROD, A.USE_SYS.

Assumptions/E.Obj	SOFT_ACS	MECH_ACS	TI_ACS	INIT_ACS	TOOLS_ACS	SAMPLE_ACS	DIS_DEV	SOFT_DLV
DLV_CONF							X	X
DLV_CONTROL							X	X
DLV_PROTECT							X	X
DLV_AUDIT							X	X
DLV_RESP							X	X
DLV_TRACE							X	X
DLV_TRANS							X	X
IC_ORG							X	
USE_TEST								X
USE_PROD							X	X
USE_SYS							X	X

Table 6. Mapping of security assumptions and objectives for the environment

O.PRODUCT_DEV is linked to A.DLV_CONTROL, A.DLV_PROTECT, A.DLV_TRANS, A.DLV_TRACE, A.DLV_AUDIT, A.DLV_RESP, A.IC_ORG and A.USE_PROD.
O_OPERATION is linked to A.USE_OPR

Assumptions/E.Obj	PRODUCT_DEV	OPERATION
DLV_CONTROL	X	
DLV_PROTECT	X	
DLV_AUDIT	X	
DLV_RESP	X	
DLV_TRACE	X	
DLV_TRANS	X	
IC_ORG	X	
USE_OPR		X
USE_PROD	X	

Table 7. Mapping of security assumptions and objectives for the environment (Table 10 continued)

5 EXTENDED COMPONENTS DEFINITION

Extended Components FCS_RND , FPT_EMSEC Definition section is based on [5]

5.1 Definition of the Family FCS_RND

To define the IT security functional requirements of the TOE a sensitive family (FCS_RND) of the Class FCS (cryptographic support) is defined here. This family describes the functional requirements for random number generation used for cryptographic purposes.
The family "Generation of random numbers (FCS_RND)" is specified as follows.

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

Component leveling:

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

Management: FCS_RND.1
There are no management activities foreseen.
Audit: FCS_RND.1
There are no actions defined to be auditable.

FCS_RND.1 Quality metric for random numbers
Hierarchical to: No other components.
Dependencies: No dependencies.

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

Following Extended Components Definition section is based on [5].

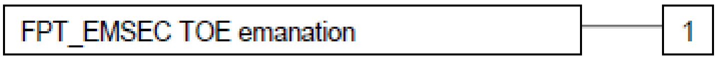
5.2 Definition of the Family FPT_EMSEC

The sensitive family FPT_EMSEC (TOE Emanation) of the Class FPT (Protection of the TSF) is defined here to describe the IT security functional requirements of the TOE. The TOE shall prevent attacks against the TOE and other secret data where the attack is based on external observable physical phenomena of the TOE. Examples of such attacks are evaluation of TOE's electromagnetic radiation, simple power analysis (SPA), differential power analysis (DPA), timing attacks, etc. This family describes the functional requirements for the limitation of intelligible emanations which are not directly addressed by any other component of CC part 2.

The family "TOE Emanation (FPT_EMSEC)" is specified as follows.

Family behavior
This family defines requirements to mitigate intelligible emanations.

Component leveling:



FPT_EMSEC.1 TOE emanation has two constituents:
FPT_EMSEC.1.1: Limit of Emissions requires to not emit intelligible emissions enabling access to TSF data or user data.

Management: FPT_EMSEC.1
There are no management activities foreseen.
Audit: FPT_EMSEC.1
There are no actions defined to be auditable.

FPT_EMSEC.1 TOE Emanation
Hierarchical to: No other components.
Dependencies: No dependencies.

FPT_EMSEC.1.1: The TOE shall not emit [assignment: *types of emissions*] in excess of [assignment: *specified limits*] enabling access to [assignment: *list of types of TSF data*] and [assignment: *list of types of user data*].

6 SECURITY REQUIREMENTS

6.1 Security Functional Requirements

This chapter defines the functional requirements for the TOE using only functional requirements components drawn from the CC version 3.1 part 2.

6.1.1 FAU Security audit

6.1.1.1 FAU_ARP Security audit automatic response

FAU_ARP.1 Security alarms

FAU_ARP.1.1 The TSF shall take **actions among the following list** upon detection of a potential security violation. [

1. **Force the card to go to death life cycle upon predefined number of unsuccessfull authentication attempts.**
2. **Force the card to go to activation life cycle if the life cycle data is corrupted and give an error to the user.**
3. **Force the card to go to a reset and clear IRAM if a tamper attack is detected by the security sensors of the IC.**
4. **Disable DF keys if DF keys are corrupted.**
5. **Give an error to the user if an uncontrolled write operation to NVM's special areas is detected.**
6. **Give a warning to the user if any corruption occurs in DFsheader, EFs header, DF PIN, DF PUK, System PIN and System PUK.]**

6.1.1.2 FAU_SAA Security audit analysis

6.1.1.2.1 FAU_SAA.1 Potential violation analysis

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

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

a) Accumulation or combination of [

1. **A wrong input of activation key while initialization and personalization keys are being loaded.**
2. **A wrong input of the current initialization and personalization key values while they are being changed.**
3. **A wrong input of initialization key while the card's NVM is being erased.**
4. **A wrong input of initialization key while the card configuration data and the application configuration data are being written to the card in the manufacturing phase.**
5. **A wrong input of cryptogram while the external interface is being authenticated by the card.**
6. **A wrong input of DF/System PIN during verify command.**
7. **A wrong input of DF/System PUK during reset retry counter command.**
8. **Integrity check of DFsheader, EFs header, when read access to the DF/EF.**
9. **Integrity check of DF PIN, DF PUK, System PIN and System PUK when verify APDU is received.**
10. **Integrity chek of life cycle data for every startup.**

]

Known to indicate a potential security violation;

b) **[None]**

6.1.2 FCS Cryptographic support

6.1.2.1 FCS_CKM Cryptographic key management

6.1.2.1.1 FCS_CKM.3 Cryptographic key access

FCS_CKM.3.1 The TSF shall perform **cryptographic key writing and reading** in accordance with a specified cryptographic key access method, **card proprietary key access functions and APDU commands** that meets the following: **none**.

6.1.2.1.2 FCS_CKM.4 Cryptographic key destruction

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method, **card proprietary key access functions and APDU commands** that meets the following: **none**.

6.1.2.2 FCS_COP Cryptographic operations

6.1.2.2.1 FCS_COP.1 Cryptographic operations

FCS_COP.1.1 Iteration 1. The TSF shall perform [**digital signature]** in accordance with a specified cryptographic algorithm [**RSA]** and cryptographic key sizes [**1024 and 2048 bits for modulus]** that meet the following: [**PKCS #1 (RSA Cryptography Standard)]**.
FCS_COP.1.1 Iteration 2. The TSF shall perform [**encryption/decryption]** in accordance with a specified cryptographic algorithm [**RSA]** and cryptographic key sizes [**1024 and 2048 bits]** that meet the following: [**PKCS #1 (RSA Cryptography Standard)]**.
FCS_COP.1.1 Iteration 3. The TSF shall perform [**encryption/decryption]** in accordance with a specified cryptographic algorithm [**DES-ECB mode]** and cryptographic key sizes [**56 bits]** that meet the following: [**DES Cryptography Standard]**.
FCS_COP.1.1 Iteration 4. The TSF shall perform [**encryption/decryption]** in accordance with a specified cryptographic algorithm [**3DES-ECB mode]** and cryptographic key sizes [**112 bits]** that meet the following: [**3DES Cryptography Standard]**.
FCS_COP.1.1 Iteration 5. The TSF shall perform [**cryptographic checksum calculation/ verification]** in accordance with a specified cryptographic algorithm [**3DES /DES]** and cryptographic key sizes [**112/56 bits]** that meet the following: [**MAC Standard]**.
FCS_COP.1.1 Iteration 6. The TSF shall perform [**encryption/decryption]** in accordance with a specified cryptographic algorithm [**AES256-ECB mode]** and cryptographic key sizes [**256 bits]** that meet the following: [**AES Standard]**.

6.1.2.3 FCS_RND Random Number Generation

FCS_RND.1 Quality metric for random numbers
FCS_RND.1.1 The TSF shall provide a mechanism to generate random numbers that meet [**FIPS-140-2**].

6.1.3 FDP: User Data Protection

6.1.3.1 FDP_ACC Access Control Policy

6.1.3.1.1 FDP_ACC.2 Complete Access control

FDP_ACC.2.1 The TSF shall enforce the **access control SFP** on **activation key, initialization key, personalization key, DF keys, DF PINs, System PIN, DF PUKs, System PUK, DFs, EFs, life cycle as objects and card activator, card initializer, card personalizer, user, authenticated user and administrator as subjects**, and all operations among subjects and objects covered by the SFP.
Reference: "Access control SFP" is described in UKIS Security Policy Model document.

FDP_ACC.2.2 The TSF shall ensure that all operations between any subject in the TSC and any object within the TSC are covered by an access control SFP.

6.1.3.2 FDP_ACF Access Control Functions

6.1.3.2.1 FDP_ACF.1 Security attribute based access control

FDP_ACF.1.1 The TSF shall enforce the **access control SFP** to objects based on the following: **activation key, initialization key, personalization key, DF PINs, System PIN, DF PUKs, System PUK, DF keys, life cycle, access permissions of DFs and EFs, error counters for CHANGE KEY, ERASE FILES, EXTERNAL AUTHENTICATE commands, resetting and error counters for DF PIN/PUK and System PIN/PUK.**
Reference: "Access control SFP" is described in UKIS Security Policy Model document.

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

- **Card activator initializes initialization key and personalization key with the activation key.**
- **Card activator activates the card with the activation key and after activation card goes to production life cycle.**
- **Card initializer formats the card with the initialization key and after format card goes to initialization life cycle.**
- **Card initializer initializes the card with the initialization key and after initialization card goes to personalization life cycle.**
- **Card initializer changes the initialization key.**
- **Card initializer erases the filesystem with the initialization key and the card goes to the production life cycle.**
- **Card personalizer personalizes the card with the personalization key and after personalization card goes to operation life cycle.**
- **Card personalizer changes the personalization key.**
- **Card personalizer erases the filesystem with the initialization key and the card goes to the production life cycle.**
- **Administrator erases the filesystem with the initialization key and the card goes to the production life cycle.**
- **Administrator creates DFs after successful system PIN authentication.**
- **Administrator changes DFs access rights after successful system PIN authentication.**
- **Administrator has all access rights on all DFs and EFs in administration phase.**

- **Administrator unblocks the card by resetting system PIN error counter with the system PUK.**
- **Administrator unblocks the DFs by resetting DF PIN error counters with system PUK.**
- **Administrator deletes EFs with CAN_NOT_DELETE_WITH_PIN access right after successful system PIN authentication.**
- **Administrator assigns/changes his own system PIN/PUK information or the authenticated user's PIN/PUK (DF).**
- **Administrator changes the life cycle (operation to administration or administration to operation) after successful system PIN authentication.**
- **Authenticated user has all access rights on DFs and EFs under a DF created with PIN except EFs with CAN_NOT_DELETE_WITH_PIN access right.**
- **Authenticated user assigns/changes his PIN/PUK information only.**
- **Authenticated user unblocks the DF by resetting DF PIN counter by DF PUK.**
- **Authenticated user writes/deletes DF keys after successful PIN authentication.**
- **User has all access rights on DFs and EFs under a DF created without PIN.**
- **User reads EFs created with READ_WITHOUT_PIN access right.**
- **User writes EFs created with WRITE_WITHOUT_PIN access right.**
- **User writes/deletes DF keys created without PIN.**]

FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: **none.**
FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the following additional rules:

- **Card initializer never changes card personalizer key.**
- **Card personalizer never changes card initializer key.**
- **Authenticated user never changes System PIN/PUK information of the smart card.**
- **User never changes System/DF PUK/PIN information.**
- **User or authenticated user never changes life cycle.**

6.1.3.3 FDP_DAU Data Authentication

6.1.3.3.1 FDP_DAU.1 Basic Data Authentication

FDP_DAU.1.1 The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of **data part of the APDU and DF/EF headers.**
FDP_DAU.1.2 The TSF shall provide **card activator, card initializer, card personalizer, user, authenticated user and administrator** with the ability to verify evidence of the validity of the indicated information.

6.1.3.4 FDP_ETC Export to outside TSF control

6.1.3.4.1 FDP_ETC.1 Export of User Data without Security Attributes

FDP_ETC.1.1 The TSF shall enforce the **Data Transmission Model** when exporting user data, controlled under the SFP(s), outside of the TSC.
Reference: "Data Transmission Model" is described in UKIS Security Policy Model document.

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

6.1.3.5 FDP_ITC Import from Outside TSF Control

6.1.3.5.1 FDP_ITC.1 Import of User Data without Security Attributes

FDP_ITC.1.1 The TSF shall enforce the **Data Transmission Model** when importing user data, controlled under the SFP, from outside of the TSC.
Reference: "Data Transmission Model" is described in UKIS Security Policy Model document.

FDP_ITC.1.2 The TSF shall ignore any security attributes associated with the user data when imported from outside the TSC.
FDP_ITC.1.3 The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TSC: **none**

6.1.3.6 FDP_RIP Residual Information protection

6.1.3.6.1 FDP_RIP.1 Subset residual information protection

FDP_RIP.1.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the **allocation of the resource from** the following objects **DF PINs, DF PUKs, DF, EF and DF keys.**

6.1.3.7 FDP_SDI Stored data integrity

6.1.3.7.1 FDP_SDI.2 Stored data integrity monitoring and action

FDP_SDI.2.1 Iteration 1. The TSF shall monitor user data stored within the TSC (TSF Scope of Control) for **memory corruption** on all objects, based on the following attributes: **EDC (Error Detection Code).**

Refinement:

Following objects are controlled with EDC:

1. **DF and EF headers**
2. **System/DF PINs and System/DF PUKs**
3. **Life cycle**
4. **DF keys**

FDP_SDI.2.2 Iteration 1. Upon detection of a data integrity error, the TSF shall **give the responses listed below.**

1. **TSF gives a warning message when DF and/or EF headers, System/ DF PIN and/or System/ DF PUK corruption is detected**
2. **TSF gives an error message and does not allow any further actions by that key when a DF key corruption is detected.**
3. **TSF forces the card back to the activation life cycle when life cycle data is corrupted.**

FDP_SDI.2.1 Iteration 2. The TSF shall monitor user data stored within the TSC (TSF Scope of Control) for **none** on all objects, based on the following attributes: **none.**

Refinement:

Following objects are not controlled for integrity errors:

1. **activation key**
2. **initialization key**
3. **personalization key**

FDP_SDI.2.2 Iteration 2. Upon detection of a data integrity error, the TSF shall **do none.**

Note: TOE has integrity error control capability for all objects except activation key, initialization key and personalization key. Since TOE assurance level EAL 4+ (AVA_VAN.5), it is sufficient to control the integrity of other objects (FDP_SDI.2 Iteration 1).

6.1.4 FIA: Identification and Authentication

6.1.4.1 FIA_AFL Authentication failures

6.1.4.1.1 FIA_AFL.1 Basic authentication failure handling

FIA_AFL.1.1 Iteration 1. The TSF shall detect when [***an administrator configurable positive integer within*** **[1 to 254]]** unsuccessful authentication attempts occur related to **verify PIN with the verify command**.
FIA_AFL.1.2 Iteration 1. When the defined number of unsuccessful authentication attempts has been ***met***, the TSF shall **forbid any access to the related DF.**

FIA_AFL.1.1 Iteration 2. FIA_AFL.1.1 The TSF shall detect when [***an administrator configurable positive integer within*** **[1 to 254]]** unsuccessful authentication attempts occur related to **verify system PIN with the verify command**.
FIA_AFL.1.2 Iteration 2. When the defined number of unsuccessful authentication attempts has been ***met***, the TSF shall **block the card.**

FIA_AFL.1.1 Iteration 3. The TSF shall detect when **[64]** unsuccessful authentication attempts occur related to [**loading of the initialization and personalization keys with Exchange Challenge command].**
FIA_AFL.1.2 Iteration 3. When the defined number of unsuccessful authentication attempts has been ***met***, the TSF shall **force the card into death life cycle.**

FIA_AFL.1.1 Iteration 4. The TSF shall detect when [**10**] unsuccessful authentication attempts occur related to **changing of the initialization and personalization keys with Change Key command, erasing of NVM with Erase Files command, and authentication of the external interface with External Authenticate command.**

Refinement: Change Key, Erase Files and External Authenticate commands use different key counters which are updated in any unsuccessfull authentication attempt to any of these commands.

FIA_AFL.1.2 Iteration 4. When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall **force the card into death life cycle.**

6.1.4.2 FIA_ATD User attribute definition

6.1.4.2.1 FIA_ATD.1 User attribute definition

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

User	Security Attributes
Card Activator	**Activation key, key error counter**
Card Initializer	**Initialization key, key error counter**
Card Personalizer	**Personalization key, key error counter**
Authenticated User	**DF PIN, PIN resetting and error counter, DF PUK, PUK error counter, DF keys, key error counter**
Administrator	**System PIN, System PIN resseting and error counter, System PUK, System PUK error counter, life cycle**

Table 8 User – Security Attributes

6.1.4.3 FIA_UAU User Authentication

6.1.4.3.1 FIA_UAU.1 Timing of authentication

FIA_UAU.1.1 The TSF shall allow **execution of Select, DIR, ReadKey, CloseFile, GetChallenge, ExchangeChallenge, Internal/external authenticate and Verify commands and execution of any operation life cycle command on a DF created without PIN,** on behalf of the user to be performed before the user is authenticated.
FIA_UAU.1.2 The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

6.1.5 FIA_UAU.4 Single-use Authentication Mechanisms

FIA_UAU.4.1 The TSF shall prevent reuse of authentication data related to **external authenticate command.**

6.1.5.1 FIA_UID User identification

6.1.5.1.1 FIA_UID.1 Timing of identification

FIA_UID.1.1 The TSF shall allow **execution of Select, DIR, ReadKey, CloseFile, GetChallenge, ExchangeChallenge, Internal/external authenticate and Verify commands and execution of any operation life cycle command on a DF created without PIN,** on behalf of the user to be performed before the user is identified.
FIA_UID.1.2 The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

6.1.5.2 FIA_USB User-subject Binding

6.1.5.2.1 FIA_USB.1 User-subject binding

FIA_USB.1.1 The TSF shall associate the appropriate user security attributes with subjects acting on behalf of that user: **activation key, initialization key, personalization key, key error counter, System/DF PIN, System/DF PIN resseting and error counter, System/DF PUK, System/DF PUK error counter, DF keys and life cycle**

6.1.6 FMT: Security management

6.1.6.1 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions: [**management of security functions, management of security attributes and management of data.]**

6.1.6.2 FMT_MOF Management of function in the TSF

6.1.6.2.1 FMT_MOF.1 Management of security functions behavior

FMT_MOF.1.1 The TSF shall restrict the ability to [***enable*]** the functions [**secure messaging]** to [**card initializer, card personalizer, user, authenticated user and administrator.]**

6.1.6.3 FMT_MSA Management of security attributes

6.1.6.3.1 FMT_MSA.1 Management of security attributes

FMT_MSA.1.1 Iteration 1. 1 The TSF shall enforce the [**UKİS access control SFP]** to restrict the ability to [***modify*]** the security attributes [**initialization key]** to **[card initializer] [refinement: in initialization phase.]**

FMT_MSA.1.1 Iteration 2. The TSF shall enforce the [**UKİS access control SFP]** to restrict the ability to [***modify*]** the security attributes [**personalization key]** to **[card personalizer] [refinement: in personalization phase.]**

FMT_MSA.1.1 Iteration 3. The TSF shall enforce the [**UKİS access control SFP]** to restrict the ability to [***modify*]** the security attributes [**PUK, life cycle]** to **[administrator] [refinement: in administration phase.]**

FMT_MSA.1.1 Iteration 4. The TSF shall enforce the [**UKİS access control SFP]** to restrict the ability to [***modify*]** the security attributes [**Secure messaging and PIN]** to **[authenticated user] [refinement: in operation phase.]**

FMT_MSA.1.1 Iteration 5. The TSF shall enforce the [**UKİS access control SFP]** to restrict the ability to [**write *and delete*]** the security attributes [**DF keys]** to **[authenticated user] [refinement: in operation phase.]**

6.1.6.3.2 FMT_MSA.2 Secure security attributes

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

1. **System/DF PIN/PUK must be minimum 4, maximum 16 characters.**
2. **DF keys can not have 0xFF value in all bytes.**

6.1.6.3.3 FMT_MSA.3 Static attribute initialization

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

1. **EFs and DFs created by FID have a default name of 0xFFs.**

2. **Maximum System/DF PIN error counter value is three if the error counter is not specified in the configuration.**
3. **Maximum DF PIN resetting counter value is 16 if the maximum reseting counter value is not specified in the configuration.**
4. **Maximum System PIN resetting counter value is 10 if the maximum system resseting counter value is not appointed in the configuration.**

Reference: “Access control SFP” is described in UKIS Security Policy Model document.

FMT_MSA.3.2 The TSF shall allow the **card initializer and card personalizer** to specify alternative initial values to override the default values when an object or information is created.
Refinement: PUK, EF and DF names’ defaut values can be overridden.

6.1.6.4 FMT_MTD Management of TSF data

6.1.6.4.1 FMT_MTD.1 Management of TSF data

FMT_MTD.1.1 Iteration 1. The TSF shall restrict the ability to **initialize** the **initialization key and personalization key** to **card activator in activation phase.**
FMT_MTD.1.1 Iteration 2. The TSF shall restrict the ability to **change** the **System/DF PIN, System/DF PIN error counter** to **administrator in administration phase.**
FMT_MTD.1.1 Iteration 3. The TSF shall restrict the ability to **read/write** the **EFs and DFs** to **authenticated user.**
FMT_MTD.1.1 Iteration 4. The TSF shall restrict the ability to **create/read/write/delete** the **EFs and DFs** to **administrator in administration phase.**

6.1.6.5 FMT_SMR Security management roles

6.1.6.5.1 FMT_SMR.1 Security roles

FMT_SMR.1.1 The TSF shall maintain the roles **card activator, card initializer, card personalizer, user, authenticated user, and administrator.**

Refinement:

1. **Card Activator: Card activator has the following capabilities defined by proving the knowledge of asymmetrical UEKAE private key only within the phases defined by TOE.**
    - **To activate the card in the activation phase with the UEKAE private key. Activation action initializes the initialization key (ba) and personalization key (ka).**

2. **Card Initializer: Card initializer has the following capabilities defined by proving the knowledge of symmetrical initialization key (ba) only within the phases defined by TOE.**
    - **To format the card in the production phase with the initialization key (ba). Format action initializes the file system and creates MF.**
    - **To write the configuration data with the initialization key (ba).**
    - **To initialize the card in the initialization phase with the initialization key (ba). Initialization action loads the data of the application that is identical for each user.**

3. **Card Personalizer: Card personalizer has the following capabilities defined by proving the knowledge of symmetrical personalization key (ka) only within the phases defined by TOE.**
   - **to personalize the card in the personalization phase with the personalization key (ka). Personalization action loads the data of the application that is distinct for each user.**

4. **User: User has the following capabilities:**
   - **to access the dedicated files which do not require PIN and access (read, write, delete) any elementary files under that dedicated files.**
   - **to access elementary files which do not require PIN.**

5. **Authenticated User: Authenticated User has the following capabilities:**
   - **to access dedicated files (DFs) which require PIN and access (read, write, delete) any elementary files under that dedicated files.**
   - **to unblock the PIN of the dedicated files (DFs) with PUK,**

6. **Administrator: Administrator has the following capabilities:**
   - **to create the dedicated files (DFs),**
   - **to delete the dedicated files (DFs),**
   - **to change the access rights(read and write) of the dedicated files (DFs),**
   - **to unblock the PIN of the dedicated files (DFs),**
   - **to change the access rights(read, write and delete) of the elemantary files (EFs),**

   to access entire files in the administration phase.

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

6.1.7 FPR: Privacy

6.1.7.1 FPR_UNO Unobservability

6.1.7.1.1 FPR_UNO.1 Unobservability

FPR_UNO.1.1 Iteration 1. The TSF shall ensure that **card personalizers** are unable to observe the operation **any** on **initialization key (ba)** by **card initializers, card activators.**
FPR_UNO.1.1 Iteration 2. The TSF shall ensure that **card initializers** are unable to observe the operation **any** on **personalization key (ka)** by **card personalizers, card activators.**
FPR_UNO.1.1 Iteration 3. The TSF shall ensure that **users** are unable to observe the operation **any** on **DFs created with PIN** by **authenticated users, administrators.**
FPR_UNO.1.1 Iteration 4. The TSF shall ensure that **users, authenticated users** are unable to observe the operation **any** on **EFs and DF's in administration life cycle** by **administrator.**

6.1.8 FPT :Protection of TOE security functions

6.1.8.1 FPT_FLS Fail secure

6.1.8.1.1 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: **card life cycle status discrepancy**, **write access out of FILE SYSTEM memory, file structure integrity failure.**

6.1.8.2 FPT_PHP TSF Physical protection

6.1.8.2.1 FPT_PHP.3 Resistance to physical attack

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

Element	**Physical tampering scenario**	**Automatic response**
PIN/PUK	**Unexpected jump in authentication code points**	**Go to a reset**
Clock	**Reduction of clock frequency to stop the TOE during a specific operation**	**Go to a reset**
Clock	**Increase clock frequency to corrupt TOE operation behavior**	**Go to a reset**
Voltage supply	**Set supply out of range voltage**	**Go to a reset**
Temperature	**Temperature out of range**	**Go to a reset**

Table 9 - Physical tampering scenarios

6.1.8.3 FPT_TDC Inter-TSF basic data consistency

6.1.8.3.1 FPT_TDC.1 Inter-TSF data consistency

FPT_TDC.1.1 The TSF shall provide the capability to consistently interpret **Key and PIN data** when shared between the TSF and another trusted IT product.

FPT_TDC.1.2 The TSF shall use the following when interpreting the TSF data from another trusted IT product:

1. Key loading (phases 4 to 6) : key data consists in a header containing the key attributes and a body containing the key value; this data block is ciphered and signed using Secure Messaging mechanism;

2. PIN loading (phases 4 to 6) : PIN data consists in a header containing the PIN attributes and a body containing the PIN value; this data block is ciphered and signed using Secure Messaging mechanism;

6.1.8.4 FPT_TST TSF self test

6.1.8.4.1 FPT_TST.1 TSF Testing

FPT_TST.1.1 The TSF shall run a suite of self tests **at the conditions:**

- **at startup**
- **to check the code memory integrity with Kart test command in all phases.**
- **to check the code memory integrity with GET DATA command and data memory integrity when the FILE SYSTEM commands are received**
- **to check write access out of FILE SYSTEM memory and unsuccessful NVM write operation when the commands that includes write operation to the FILE SYSTEM are received**
- **to check defined Card Life cycle at Power On state**
- **during cryptographic operations**

to demonstrate the correct operation of **the TSF.**

FPT_TST.1.2 The TSF shall provide authorized users with the capability to verify the integrity of **the TSF data.**
FPT_TST.1.3 The TSF shall provide authorized users with the capability to verify the integrity of **the TSF**.

6.1.8.5 FPT_EMSEC.1 TOE Emanation

FPT_EMSEC.1.1 The TOE shall not emit **power variations, electromagnetic emission variations and timing variations during command execution** in excess of **non-useful information** enabling access to **Initialization, personalization and DF keys** and [**none**].

6.2 Security Assurance Requirements for the TOE

The security assurance requirement level is EAL 4 augmented (AVA_VAN.5).

6.3 Security Requirements Rationale

Each of the security objectives for the environment during the development phase is addressed by at least one assurance requirement.

6.3.1 Security Assurance Requirements meet Security Objectives for the environment

6.3.1.1 Life Cycle Phase 1 (development phase)

This section demonstrates that the combination of the Assurance components is suitable to satisfy the identified security objectives for the environment during the development phase.
Each of the security objectives for the environment is addressed by assurance components.
The following table (Table 10) demonstrates which Assurance component contribute to the satisfaction of each security objective for the environment. For clarity, the table does not identify indirect dependencies.

Assurance Componenets /E.Obj	SOFT_ACS	MECH_ ACS	TI_ACS	INIT_ACS	TOOLS_ ACS	SAMPLE_ACS
ALC_DVS.1	X	X	X	X	X	X

Table 10. Mapping of assurance components and security objectives for the environment during the development phase.

The assurance component ALC_DVS.1 measures are designed to meet access objectives and specifically O.SOFT_ACS, O.MECH_ACS, O.TI_ACS, O.INIT_ACS, O.TOOLS_ACS and O.SAMPLE_ACS.

6.3.1.2 TOE Life Cycle Phases (Ref: ST Figure 3) except activation phase

The assurance measures related with AGD_ADM.1 and AGD_USR.1 meet O.Operation environmental objective. Assurance measures canalizes the users and administrators when TOE gives a warning message related with corrupted objects.

6.3.2 Security Functional Requirements Rationale

This section demonstrates that the combination of the security requirement objectives is suitable to satisfy the identified IT security objectives.
Each of the IT security objectives is addressed by functional requirements.
The following table (Table 11) demonstrates which functional requirements contribute to the satisfaction of each security objective for the TOE. For clarity, the table does not identify indirect dependencies.
This section describes why the security requirements are suitable to provide each of the IT security objectives.

SFR/T.OBJ	INTEGRITY	FUNCTION	OPERATE	FLAW	DIS_MECHANISM	DIS_ MEMORY	MOD_ MEMORY	CLON	PROT_INF_ LEAK	PROT_PHYS_ TAMPER
EAL 4+ requirements				X						
FAU_ARP.1	X				X	X	X			
FAU_SAA.1	X				X	X	X			
FCS_CKM.3						X		Partial		X
FCS_CKM.4						X		Partial		X
FCS_COP.1 (I.1, I.2, I.3, I.4, I.5,I.6)						X		Partial		X
FCS_RND.1						X	X	Partial		
FDP_ACC.2		X	Partial		X	X	X	Partial		X
FDP_ACF.1		X			X	X	X	Partial		X
FDP_DAU.1	X		Partial				X	Partial		X
FDP_ETC.1						X		Partial		
FDP_ITC.1	X						X			
FDP_RIP.1						X		Partiall		X
FDP_SDI.2 (I.1)	X		Partial				X	Partial		
FDP_SDI.2 (I.2)	X		Partial							
FIA_AFL.1 (I.1, I.2, I.3, I.4)			X					Partial		X
FIA_ATD.1								Partial		X
FIA_UAU.1						X	X	Partial		X
FIA_UAU.4						X	X	Partial		
FIA_UID.1						X	X	Partial		X
FIA_USB.1						X	X	Partial		X
FMT_SMF.1			X		X	X	X	X		X
FMT_MOF.1			X		X	X	X			X
FMT_MSA.1 (I.1, I.2, I.3, I.4, I.5)					X	X	X	Partial		X
FMT_MSA.2										X
FMT_MSA.3					X	X	Partial	Partial		X
FMT_MTD.1 (I.1, I.2, I.3, I.4)					X	X	X			X
FMT_SMR.1			X							X

FPR_UNO.1 (I.1, I.2, I.3, I4)					X						
FPT_FLS.1			X						X	X
FPT_PHP.3									X	X
FPT_TDC.1	X									
FPT_TST.1	X								X	
FPT_EMSEC.1									X	

Table 11. Mapping of security functional requirements and IT objectives

The EAL 4+ assurance requirements contribute to the satisfaction of the O.FLAW security objective. They are suitable because they provide the assurance that the TOE is designed, implemented and operates so that the IT functional requirements are correctly provided.

Security audit functional requirements FAU_ARP.1 and FAU_SAA.1 detect security violating actions such as integrity loss (corresponding to the security objective O.INTEGRITY), and actions which could disclose security mechanisms (corresponding to the security objective O.DIS_MECHANISM), stored memory (corresponding to the security objective O.DIS_MEMORY), or modification of stored information (corresponding to the objective O.MOD_MEMORY).

Cryptographic support functional requirements: FCS_CKM.3, FCS_CKM.4 and FCS_COP.1 (I1, I2, I3, I4, I5, I6) support the access control to the assets. These functions cooperate to meet the security objectives of O.DIS_MEMORY, and thus participate to meet the O.CLON security objective.

Access control functional requirements FDP_ACC.2 and FDP_ACF.1 control the access conditions. This fulfills the security objectives, O.FUNCTION, O.DIS_MECHANISM (Code), O.DIS_MEMORY and O.MOD_MEMORY (Data). They participate to the fulfillment of O.CLON.

FDP.ACC.2 contributes to the correct operation of the TOE (corresponding to the security objectives O.OPERATE. and O.CLON).

Data authentication functional requirement FDP.DAU.1 assures the objectives O.INTEGRITY, and O.MOD_MEMORY by verifying the evidence of validity of the data. It contributes to the correct operation of TOE, corresponding to the objective O.OPERATE and makes cloning more difficult, O.CLON.

Export to outside TSF control function FDP_ETC.1 contributes to realization of O.DIS_MEMORY by controlling the export of user data. It contributes to the correct operation of TOE, O.CLON.

Import from outside TSF control function FDP_ITC.1 contributes to realization of O.MOD_MEMORY by controlling the import of user data. This also contributes to O.INTEGRITY.

FDP_RIP.1 functional requirement meets O.DIS_MEMORY objective by assuring that previous information cannot be used out of context.. It also contributes to the correct operation of TOE, O.CLON which relies on disclosure of confidential information.

FDP_SDI.2 (I1) functional requirement meets O.INTEGRITY, and O.MOD_MEMORY objectives by detecting and acting on integrity errors. It also contributes to the correct operation of TOE, corresponding to the security objective O.OPERATE and O.CLON.

Identification and authentication functional requirements FIA_AFL.1 (I1, I2, I3, I4) meets the security objective O.OPERATE. It also contributes to the correct operation of TOE, O.CLON.

Identification and authentication functional requirements FIA_UAU.1 meets O.DIS_MEMORY and O.MOD_MEMORY objectives by managing the authentication of candidates. All of them also contribute to the correct operation of TOE, O.CLON.

Identification and authentication functional requirement FIA_UAU.4 and FCS_RND.1 (for random number generation during authentication) prevents an unauthorized access to stored memory, and thus contributes to fulfilling the security objectives O.DIS_MEMORY and O.MOD_MEMORY. It also contributes to the correct operation of TOE, O.CLON.

Identification and authentication functional requirements FIA_UID.1 and FIA_USB.1 meet O.DIS_MEMORY and O.MOD_MEMORY objectives by use of identification and by binding it to the subject. They also contribute to the correct operation of TOE, O.CLON.

FMT_SMF functional requirement meets O_OPERATE, O_DIS_MECHANISM, O.DIS_MEMORY, O.MOD_MEMORY, O_CLON objectives by performing management of security attributes and data.

FMT_MOF.1 functional requirement meets O.OPERATE, O.DIS_MECHANISM, O.DIS_MEMORY and O.MOD_MEMORY objectives by managing the security functions which fulfill these objectives.

FMT_MSA.1 (I1, I2, I3, I4, I5) and FMT_MSA.3 meet O.DIS_MECHANISM, O.DIS_MEMORY and O.MOD_MEMORY objectives. They also contribute to the correct operation of TOE, O.CLON.

FMT_MTD.1 (I1, I2, I3, I4) functional requirement meets O.DIS_MECHANISM, O.DIS_MEMORY and O.MOD_MEMORY objectives by management of TSF data.

FMT_SMR.1 functional requirement meets O.OPERATE objective due to role management.

Unobservability requirement FPR_UNO.1 (I1, I2, I3, I4) assures that unauthorized parties cannot over look settings of cards security mechanisms, corresponding to the security objective O.DIS_MECHANISM.

FPT_FLS.1 functional requirement meets O.OPERATE and O.PROT_INF_LEAK, objectives by assuring secure state when failures occur (intentional or not).

FPT_TDC.1 functional requirement meets O.INTEGRITY objective by assuring inter TSF data consistency.

FPT_TST.1 functional requirement meets O.INTEGRITY and O.PROT_INF_LEAK objective by detecting non integrity during self tests.

FPT_PHP.3 functional requirement meets O.PROT_INF_LEAK and O.PROT_PHYS_ TAMPER.

O.PROT_INF_LEAK requires the TOE to protect confidential TSF data stored and/or processed in the chip against disclosure which is addressed by the FPT_EMSEC.1, FPT_FLS.1, FPT_TST.1, and FPT_PHP.3.

6.3.3 Dependencies of security requirements

This section is intended to be a demonstration that the dependencies between the security requirements components (functional and assurance) included in this ST are satisfied.

Assurance requirements specified in this ST are precisely as defined in EAL4 with one higher hierarchical component (AVA_VAN.5). AVA_VAN.5 has no dependencies.

EAL 4 is asserted to be a known set of assurance components for which all dependencies are satisfied.

The following table (Table 12) lists all functional requirements components including security requirements on the IT environment. For each component, the dependencies specified in Common Criteria are listed, and a reference to the component number is given.

Number	Security functions	Dependencies
1	FAU_ARP.1 :Security Alarms	FAU_SAA.1
2	FAU_SAA.1 : Potential Violation Analysis	**FAU_GEN.1 (*)**
3	FCS_CKM.3 : Cryptographic Key Access	[FDP_ITC.1 or FCS_CKM.1 or FDP_ITC.2] FCS_CKM.4
4	FCS_CKM.4 : Cryptographic Key Destruction	[FDP_ITC.1 or FCS_CKM.1 or FDP_ITC.2]
5	FCS_COP.1 : Cryptographic Operation	[FDP_ITC.1 or FCS_CKM.1 or FDP_ITC.2] FCS_CKM.4
6	FCS_RND.1: Random number generation	No dependencies
7	FDP_ACC.2 : Complete Access Control	FDP_ACF.1
8	FDP_ACF.1 : security attributes based Access Control Functions	FDP_ACC.1 FMT_MSA.3
9	FDP_DAU.1 : basic Data Authentication	No dependencies
10	FDP_ETC.1 : Export of user data without security attributes	FDP_ACC.1 or FDP_IFC.1
11	FDP_ITC.1 : Import of user data without security attributes	FDP_ACC.1 or FDP_IFC.1 FMT_MSA.3
12	FDP_RIP.1 : subset residual information protection	No dependencies
13	FDP_SDI.2 : stored data integrity monitoring and action	No dependencies
14	FIA_AFL.1 : basic authentication failure handling.	FIA_UAU.1
15	FIA_ATD.1 : user attribute definition	No dependencies
16	FIA_UAU.1 : timing of authentication	FIA_UID.1
17	FIA_UAU.4 : Single-use authentication mechanisms	No dependencies
18	FIA_UID.1 : timing of identification	No dependencies
19	FIA_USB.1 : user-subject binding	FIA_ATD.1
20	FMT_MOF.1 : management of security functions behavior	FMT_SMR.1, FMT_SMF.1
21	FMT_MSA.1 : management of security attributes	[FDP_ACC.1 or FDP_IFC.1] FMT_SMR.1, FMT_SMF.1
22	FMT_MSA.2 : safe security attributes	FDP_ACC.1 or FDP_IFC.1 FMT_MSA.1 FMT_SMR.1
23	FMT_MSA.3 : safe attributes initialization	FMT_MSA.1 FMT_SMR.1
24	FMT_MTD.1 : management of TSF data	FMT_SMR.1 , FMT_SMF.1
25	FMT_SMR.1 : security roles	FIA_UID.1
26	FMT_SMF.1 : specification of management functions	No dependencies
27	FPR_UNO.1 : Unobservability	No dependencies
28	FPT_FLS.1 : failure with preservation of secure state	No dependencies
29	FPT_PHP.3 : Resistance to physical attacks	No dependencies
30	FPT_TDC.1 : inter-TSF basic TSF data consistency	No dependencies
31	FPT_TST.1 : TSF testing	No dependencies
32	FPT_EMSEC.1: TOE Emanation	No dependencies

Table 12. Dependencies analysis

* Dependencies not met for reasons given below.

The following dependencies marked by "*" in Table 12 are not applicable to the TOE security functional requirements:

FDP_ACC.2 is hierarchical to FDP_ACC.1, therefore dependencies on FDP_ACC.1 can be met by FDP_ACC.2.

FAU_GEN.1 is not applicable to the TOE: Indeed if FAU_GEN.1 is chosen in the ST, it forces many security relevant events to be recorded, and this is not applicable to the smartcard as many of these events bring the card to an insecure state where recording itself could open a security breach.

6.4 Evaluation assurance level rationale

An assurance requirement of EAL4 is required for this type of TOE since it is intended to defend against sophisticated attacks. In order to provide a meaningful level of assurance that the TOE provides an adequate level of defense against such attacks, the evaluators should have access to the low level design and source code. The lowest for which such access is required is EAL4. The assurance level EAL4 is achievable, since it requires no specialist techniques on the part of the developer.

AVA_VAN.5: Advanced methodical vulnerability analysis
Because vulnerability analysis is very important issue for this type of TOE, the highest level of AVA_VAN which is AVA_VAN5 is chosen. AVA_VAN5 has no dependencies.

6.5 Security requirements are mutually supportive and internally consistent.

The purpose of this part of the ST rationale is to show that the security requirements are mutually supportive and internally consistent.
EAL4 is an established set of mutually supportive and internally consistent assurance requirements.
The dependencies analysis for the additional assurance component in the previous section has shown that the assurance requirements (EAL 4 assurance requirements and AVA_VAN.5) are mutually supportive and internally consistent (all the dependencies have been satisfied).
The dependencies analysis for the functional requirements described above demonstrate mutual support and internal consistency between the functional requirements.

6.5.1 Rationale that Requirements are Mutually Supportive

The security requirements work mutually so that each SFR is protected against bypassing, tampering, deactivation and detection attacks by other SFRs.

6.5.1.1 Bypass

Prevention of bypass is derived as described below:

FIA_UID.1 and **FIA_UAU.1** support other functions' allowing user access to data by limiting the actions the user can take prior to identification and authentication.

FIA_UAU.4, FCS_RND.1 prevents reuse of authentication data, thus reducing the probability of bypass.

The management functions, including **FMT_MOF.1**, **FMT_MSA.1**, and **FMT_MTD.1** support all other SFRs by restricting the ability to change certain management functions to certain specified roles, thus ensuring that other users cannot circumvent these SFRs.
FMT_MSA.2 and **FMT_MSA.3** limit the acceptable values for secure data, thus providing protection from bypass to those SFRs dependent on that data.

6.5.1.2 Tamper

Prevention of tamper is derived as described below:

FCS_CKM.3, **FCS_CKM.4** and **FCS_COP.1** provide for the secure handling of keys, and therefore support those SFRs that may rely on the use of those keys.
FIA_UID.1 and **FIA_UAU.1** support other functions allowing user access to data by limiting the actions the user can take prior to identification and authentication.
FIA_UAU.4 prevents reuse of authentication data, thus reducing the probability of tamper.
The management functions, including **FMT_MOF.1, FMT_MSA.1**, and **FMT_MTD.1** support all other SFRs by restricting the ability to change certain management functions to certain specified roles, thus ensuring that other users cannot circumvent these SFRs.
FMT_MSA.2 and **FMT_MSA.3** limit the acceptable values for secure data, thus providing protection from tampering to those SFRs dependent on that data.
FPT_PHP.3 supports physical tampering protection by using the data which is taken by physical sensors.

6.5.1.3 Deactivation

Prevention of deactivation is derived as described below:

The access control SFP detailed in **FDP_ACF.1** (Reference: “Access control SFP” is described in UKIS Security Policy Model document.) along with the other SFRs dealing with Access control, provide for rigorous control of allowed data manipulations and thus prevent unauthorized deactivation.

The management functions, including **FMT_MOF.1**, **FMT_MSA.1**, and **FMT_MTD.1**, support all other SFRs by restricting the ability to change certain management functions to certain specified roles, thus ensuring that other users cannot circumvent these SFRs.
FMT_MSA.2 and **FMT_MSA.3** limit the acceptable values for secure data, thus providing protection from deactivation to those SFRs dependent on that data.

6.5.1.4 Detection

Detection is derived as described below:
The management functions, including **FMT_MOF.1, FMT_MSA.1**, and **FMT_MTD.1**, support all other SFRs by restricting the ability to change certain management functions to certain specified roles, thus ensuring that other users cannot circumvent these SFRs.

FMT_MSA.2 and **FMT_MSA.3** limit the acceptable values for secure data, thus providing detection protection to those SFRs dependent on that data.

FPT_PHP.3 supports detection by using the data which is taken by physical sensors.

FDP_SDI.2 (Iteration 1) supports detection.

7 TOE Summary Specification

7.1 TOE Security Functions

7.1.1 Cryptographic Operations

7.1.1.1 Sign

In Sign security function, plain data sent by the user within the APDU command is signed (decrypted) with the key that is previously referenced with another command. Signed data is transmitted back to the user. The point here is not the secrecy of the data; it is the integrity of the data. RSA (1024 or 2048) algorithm can be used for this operation, so the referenced key must be an RSA 1024 or 2048 key and it must own all the parameters required for this operation.

Note: This function meets the following TOE security functional requirement:

- *FCS_COP.1 (Iteration 1).*

7.1.1.2 Verify Signature

In Verify Signature security function, signed part of the data sent by the user within the APDU command is encrypted with the key that is previously referenced with another command and the encrypted data is compared with the plain part of the data sent at the end of signed data within the command. After the comparison, a response is transmitted back to the user indicating whether the signature is verified or not. The point here also is not the secrecy of the data; it is the integrity of the data. RSA (1024 or 2048) algorithm can be used for this operation, so the referenced key must be an RSA 1024 or 2048 key and it must own all the parameters required for this operation.

Note: This function meets the following TOE security functional requirement:

- *FCS_COP.1 (Iteration 1).*

7.1.1.3 Encryption

In Encryption security function plain data sent by the user within the APDU command is encrypted with the key that is previously referenced with another command. Encrypted data is transmitted back to the user as a response. Here both the secrecy and the integrity of the data is of concern. For the encryption operation, any of the RSA 1024, RSA 2048, DES3-ECB, DES-ECB and AES-ECB algorithms can be used, so the referenced key can be any of these algorithms' keys. But the key must own all the parameters required for this operation.

Note: This function meets the following TOE security functional requirement:

- *FCS_COP.1 (Iteration 2, 3, 4, 6).*

7.1.1.4 Decryption

In Decryption security function, cipher data sent within the APDU command is decrypted with the key that is previously referenced with another command. The plain text is transmitted back to the user as a response. Also here both the secrecy and the integrity of the data is of concern. For the decryption operation, any of the RSA 1024, RSA2048, DES3-ECB, DES-ECB and AES-ECB algorithms can be used, so the referenced key can be any of these algorithms' keys. But the key must own all the parameters required for this operation.

For the correct operation of the security functions described above, the user should reference an appropriate key (application-DF key) before the cryptographic operation takes place. Here to reference a key means moving the key from the NVM memory area into the RAM memory area in order to use it. Also before this operation, the user must load the key into that application specific NVM memory area in a secure way. Loading more than 1 key to an application (DF) is possible (maximum 20 keys). The algorithms for these keys may be different (any of RSA1024, RSA 2048, DES3-ECB, DES-ECB and AES-ECB).

Note: This function meets the following TOE security functional requirement:

- *FCS_COP.1 (Iteration 2, 3, 4, 6).*

7.1.1.5 Cryptographic Checksum Calculation

Cryptographic checksum is used in order to protect user data integrity. Cryptographic checksum calculation function calculates the checksum of the plain data and the initialization vector sent within the command according to the reference key sent prior to the command by the user.

The first part of the plain data sent within the command is XORed with the initialization vector and encrypted with the reference key. The data formed after this operation serves as the new initialization vector for the second part of the plain data. The operation is repeated until all parts of the data is encrypted. Calculation of cryptographic checksum is performed using DES or 3DES algorithms in the TOE. That's why, the reference key must belong to one of these algorithms.

Note: This function meets the following TOE security functional requirement:

- *FCS_COP.1 (Iteration 5).*

7.1.1.6 Cryptographic Checksum Verification

Cryptographic checksum verification is performed in two steps. Firstly, the checksum of the plain data and the initialization vector sent within the command is calculated according to the reference key. Secondly, the calculated checksum is compared with the checksum within the command. If they match, an operation successfull response is returned. If they don't match, an error message is returned. A mismatch means that the data integrity has been corrupted. Calculation of cryptographic checksum is performed using DES or 3DES algorithms in the TOE. That's why, the reference key must belong to one of these algorithms.

Note: This function meets the following TOE security functional requirement:

- *FCS_COP.1 (Iteration 5).*

7.1.2 Authentication and Authorization Functions

7.1.2.1 Administrator Authentication (with System PIN)

Administration life cycle is a life cycle which allows only the administrator to run administration commands. In order to pass to the administration life cycle, System PIN must be verified. If a wrong System PIN is entered 3 times, the card is blocked. Card can be unblocked using System PUK. Only the administrator can change the System PIN. System PIN must be minimum 4, maximum 16 digits. System PIN intial value is given at production state during constitution of MF. Maximum System PIN resetting counter value is 10 if the maximum system resseting counter value is not appointed in the configuration.

Administration commands such as CHANGE_KEY, ERASE_FILES are to be performed in this life cycle by the administrator (after personalization phase). Only administrator can change the life cycle from Operation to Administration and vice versa.

Note: This function meets the following TOE security functional requirements:

- *FAU_ARP.1*
- *FAU_SAA.1,*
- *FDP_ACC.2, FDP_ACF.1,*
- *FIA_ATD.1, FIA_AFL.1(Iteration 1,2,3,4), FIA_USB.1,*
- *FMT_MSA.1(Iteration 3,4), FMT_MSA.2, FMT_MSA.3, FMT_MTD.1 (Iteration 2), FMT_SMR.1, FMT_SMF.1,*
- *FPR_UNO.1.(Iteration 1,2,3,4)*

7.1.2.2 Authenticated User Authentication (with PIN)

On a directory (DF) created with PIN, in order to perform PIN verification in operation life cycle, PIN must be set first.

PIN must be minimum 4, maximum 16 digits. When the PIN is input maximum PIN error value times (if it is not set at configuration, default value is 3) incorrectly, that directory (DF) becomes INVALID and only the administrator can make that DF reusable by resetting PIN error counter. After the error counter is reset, authenticated user can use his DF with the PIN the administrator gave him. During PIN change, if the old PIN is input incorrectly, error counter is incremented by 1. After maximum PIN retry number incorrect entries, the DF becomes INVALID. Maximum DF PIN resetting counter value is 16 if the maximum reseting counter value is not specified in the configuration.

For performing operations in operation life cycle on a DF created with PIN, VERIFY command must be performed successfully. Access to the EFs/DFs under that DF is dependent to their own access conditions (Table 13).

Operation life cycle is a life cycle belonging to the user and the authenticated user usually. For this reason, in order to change the life cycle to administration, system PIN must be entered. Furthermore, a user/authenticated user can not create directories (DFs).

Note: This function meets the following TOE security functional requirements:

- *FAU_SAA.1,*
- *FDP_ACC.2, FDP_ACF.1,*
- *FIA_ATD.1, FIA_AFL.1(Iteration 1,2,3,4), FIA_USB.1, FIA_UAU.1, FIA_UID.1,*
- *FMT_MSA.1(Iteration 5), FMT_MSA.2, FMT_MSA.3, FMT_MTD.1 (Iteration 3,4), FMT_SMR.1, FMT_SMF.1,*
- *FPR_UNO.1(Iteration 1,2,3,4)*

7.1.2.3 Authorizing User to an Operation

Authorizing user to an operation function is used for making the decision if the user is authorized or not to perform the operation he wants. In this function, the user must transmit a secret data known both by the user and the TOE within the operation's command. The user is authorized to the operation only if he transmits that secret data accurately and completely. Otherwise, the user will not be allowed to perform that operation. Here, the secret data transmitted in the command can be a key encrypted by itself or a special data encrypted by a key depending on the involved command and the user type.

For being authorized, the user should either know the key or both the key and the special data according to the command and the user type. For this operation, one of RSA 1024 or 2048 and DES-ECB or DES3-ECB algorithms can be used depending on the command being used. So the referenced key may belong to one of these algorithms, but the key must own all the parameters required for this operation.

This function is concerned with the commands; Exchange Challenge for activation (RSA), Change Key (DES3-ECB), Erase Files (DES3-ECB) and External Authenticate (DES3-ECB/ DES-ECB). There is an error counter for each of these commands seperately. The secret data used for authorization may be common for some of these commands, but the error counter is not counted for each faulty usage of the secret data itself, it is counted for each faulty usage of the command.

Note: This function meets the following TOE security functional requirements:

- *FAU_SAA.1,*
- *FDP_ACC.2, FDP_ACF.1,*
- *FIA_ATD.1, FIA_AFL.1(Iteration 1,2,3,4), FIA_USB.1,*
- *FMT_MSA.1(Iteration 1,2), FMT_MTD.1(Iteration 1), FMT_SMF.1,*
- *FPR_UNO.1. (Iteration 1,2,3,4)*

7.1.2.4 Authentication of User to TOE

Authentication of user to TOE function is used to determine if the user is a secure user in order to use the active application. User is expected to transmit a secret data known both by the user and the application within the EXTERNAL AUTHENTICATE command. If the user transmits this secret data correctly and completely, he is defined as a secure user for the application. Otherwise, the user will not be allowed to perform any secure operation on that application. Here, the secret data transmitted within the command is a random number generated by the TOE and encrypted with a key belonging to that application. Each random generated by the TOE can only be used once. TOE guarantees a random number to be used for the authentication of user to TOE at most once. For this operation, one of DES3-ECB and DES-ECB algorithms can be used, so the referenced key may belong to one of these algorithms, but the key must own all the parameters required for this operation.

Note: This function meets the following TOE security functional requirements:

- *FCS_RND.1*
- *FDP_ACC.2, FDP_ACF.1,*
- *FIA_UAU.4.*

7.1.2.5 Authentication of TOE to User

Authentication of TOE to user function is used to decide whether the TOE is secure and correct TOE or not. TOE is expected to encrypt the data within the incoming command with a key known both by the user and the TOE and transmit back the encrypted data. TOE is defined as a secure TOE for the user, only if transmits this secret data correctly and completely. Otherwise, it is not reliable for a user to use the TOE. Here, the secret data transmitted within the response is a random number generated by the user and encrypted with a key belonging to that application. For this operation, one of DES3-ECB and DES-ECB algorithms can be used, so the referenced key may belong to one of these algorithms, but the key must own all the parameters required for this operation.

Note: This function meets the following TOE security functional requirement:

- *FPT_TDC.1*
- *FCS_RND.1*

7.1.3 Cryptographic Keys

Crytographic key management is performed within the TOE. Key selection and controls (check operations regarding whether the key content is appropriate or not), key destruction are operations performed by the TOE.

Note: This function meets the following TOE security functional requirements:

- *FAU_ARP.1,*
- *FCS_CKM.3, FCS_CKM.4,*
- *FDP_ACC.2, FDP_ETC.1, FDP_ITC.1,*
- *FIA_ATD.1,*
- *FMT_MSA.1(Iteration 5), FMT_MSA.2, FMT_SMF.1*
- *FDP_RIP.1*

7.1.4 Secure Messaging

With a bit in APDU command's CLA byte, it is decided whether to use secure (encrypted) messaging or not. Secure messaging is mandotory according to MF or DF access rights. If MF is created with secure messaging access right all commands under MF must be encrypted. If DF is created with secure messaging access right all commands under DF must be encrypted. EXCHANGE CHALLENGE command does not need to be encrypted. In secure messaging all the data transmitted within a command is encrypted with the session key according to DES3-ECB algorithm. As both sides (users and TOE) know the session key, they decrypt the incoming commands with the session key to interpret them.

Note : This function meets the following TOE security functional requirement :

- *FDP_ETC.1, FDP_ITC.1*
- *FMT_MOF.1.*
- *FCS_RND.1*

7.1.5 Integrity of the Objects

DF, DF keys, EF, System/DF PIN, System/DF PUK and life cycle integrity check is peformed with checksums. In every write and erase operation the checksum is being updated, checksum is controlled in every read operation. If there is a corruption in DF, EF, System/DF PIN, and System/DF PUK, a warning or error message is returned as a response to the user. If there is a corruption in DF keys, System/DF PIN, System/DF PUK an error is returned and the corrupted data is no longer allowed to use. If there is a corruption in EF header, an error is returned and the corrupted EF is no longer allowed to use. But if the corruption occurred in EF body a warning returned and the corrupted EF is used. When DF, EF, DF PIN/PUK or DF keys are being deleted, the delete operation is performed by releasing the connections on tables. When a page is taken from the memory area, that page is being erased before the operation.

During DF/EF creation and System/DF PIN/PUK change if the operation is interrupted by ejecting the card from the card reader, old data becomes valid in order to protect the data integrity, because it is not clear where the write operation is interrupted.

When an uncontrolled access is detected to write to the special areas of NVM, an error code indicating a memory error is returned and the write operation is not permitted.

Command integrity is provided with the checksum byte which is at the end of the command. If the checksum is wrong, the command sent from card to the reader or command sent from the reader to the card must be repeated.

At each reset, card life cycle is tested whether it is one of the defined states or not, checksum is also controlled. If the card life cycle data is corrupted, TOE returns to the activation life cycle.

Code memory checksum is calculated and returned to the user within the GET DATA command. User can check the code memory integrity by this way.

Self tests are configured by the TOE.

Activation key, initialization key, and personalization key are not controlled for integrity errors.

Note: This function meets the following TOE security functional requirements:

- *FAU_ARP.1,*
- *FDP_ACC.2, FDP_DAU.1, FDP_SDI.2(Iteration 1,2),*
- *FPT_FLS.1, FPT_TST.1.*
- *FAU_SAA.1*
- *FDP_ITC.1*

7.1.6 Access Conditions on the DFs and EFs

DF/EF access conditions are controlled according to the command to be performed. Access control is made:

- Read/Write access: If the DF has a write access, new DFs and EFs can be created/deleted under that DF. If the EF has a write access, EF can be written/updated. In order to read an EF, the EF must have read access.
- Security access with System/DF PIN: User can access DF and any EF under that DF if the DF is created without PIN. If the DF is created with PIN, access conditions of EFs under that DF, depends on the access conditions of EF in the operation life cycle (Table 13)

DF with key	DF with PIN	EF Read Without PIN	EF Write Without PIN	EF Read Without Auth.	EF Write Without Auth.	EF can not Delete With PIN	Result
-	-	-	-	-	-	-	Uncontroled read/write, delete
	X	-	-				Read/write with PIN authentication
	X	X	-				Write with PIN, uncontrolled read
	X	-	X				Read with PIN, uncontrolled write
	X	X	X				Read/write uncontrolled
X	-	-	-	X			Read with key authentication, write uncontrolled
					X		Write with key authentication, read uncontrolled
				X	X		Read/write with key authentication
	X					X	File can not delete with PIN authentication
	X						File is deleted with PIN authentication

Table 13. Access conditions for TOE EFs/DFs

- Security access with key authentication: If the DF is created with key authentication, the user uses the internal and external authenticate commands in order to get authenticated into that directory. This subject is explained in Authentication of User to TOE.

Note : This function meets the following TOE security functional requirements:

- *FDP_ACF.1 ,FDP_ACC.2,*
- *FMT_MTD.1(Iteration 3,4), FMT_SMR.1, FMT_SMF.1*
- *FDP_ETC.1 ve FDP_ITC.1*
- *FDP_RIP.1*

7.1.7 Function Countering Physical Attacks

This function protects the TSF functionality, TSF data and user data. It implements the following protection operations for the TOE:

1. Hiding information about IC power consumption, electromagnetic emenation and command execution time: *FPT_EMSEC*
2. Detection of the physical tampering of the TSF with sensors for operating voltage, clock frequency, temperature. If the TOE detects with the mentioned sensors, that it is not supplied within the specified limits, a security reset is initiated and the TOE is not operable until the supply is back in the specified limits. The hardware protects itself against analyzing and physical tampering: *FPT_TST*, *FAU_ARP.1*, *FPT_PHP.3*
3. Make redundant code execution against fault attacks to the cryptographic operations and other critical code executions such as "pin verify" operation: *FPT_PHP.3*
4. Runs random number generator tests during initial start-up and at any cryptographic operations. Preserve a secure state when a failure is detected by TSF according to *FPT_TST.1*.
5. Provide the capability to verify the integrity of ROM at the initialization phase and other life cycle phases of the TOE. *FPT_TST.1*

6. Verifies the integrity of the keys, header data of DFs, header data of EFs, pins and puks. If the integrity check of keys is failed keys can not be used, if header data of EFs and DFs fails, an error is returned to the user. If an integrity check of interior filesystem tables fails, the card goes to a reset. *FPT_TST.1, FAU_ARP*.

- *FPT_PHP.3*
- *FAU_ARP.1*
- *FPT_TST.1*
- *FPT_EMSEC*

7.2 TOE Summary Specification Rationale

7.2.1 Cryptographic Operations

7.2.1.1 Sign

Sign function implements digital signature operation described in FCS_COP.1 (Iteration 1).

7.2.1.2 Verify Signature

Verify signature function implements signature verification operation described in FCS_COP.1 (Iteration 1).

7.2.1.3 Encryption

Encryption function implements encryption operation described in FCS_COP.1 (Iteration 2,3,4,6).

7.2.1.4 Decryption

Decryption function implements decryption operation described in FCS_COP.1 (Iteration 2,3, 4,6).

7.2.1.5 Cryptographic Checksum Calculation

Cryptographic checksum calculation is described in FCS_COP.1 (Iteration 5).

7.2.1.6 Cryptographic Checksum Verification

Cryptographic checksum verification is described in FCS_COP.1 (Iteration 5).

7.2.2 Authentication and Authorization Functions

7.2.2.1 Administrator Authentication (with System PIN)

- FAU_ARP.1 unsuccessfull authentication,
- FAU_SAA.1 monitoring System PIN verification,
- FDP_ACC.2 and FDP_ACF.1 access control SFP enforcement on System PIN,
- FIA_ATD.1 maintaining System PIN as administrator security attribute,
- FIA_AFL.1 (Iteration 1,2,3,4) handling unsuccessfull System PINauthentication,
- FIA_USB.1 binding administrator withSystem PIN,
- FMT_SMF.1 specification of management functions,
- FMT_MSA.1 (Iteration 3,4), FMT_MSA.2, FMT_MSA.3 defining restrictive maximum System PIN error counter and maximum reseting number,
- FMT_MTD.1 (Iteration 2) change DF PIN,
- FMT_SMR.1 maintaining the administrator role,
- FPR_UNO.1 unobservability of administrator is implemented by this function. (Iteration 1,2,3,4)

7.2.2.2 Authenticated User Authentication (with PIN)

- FAU_SAA.1 monitoring DF PIN verification,
- FDP_ACC.2 and FDP_ACF.1 access control SFP enforcement on DF PIN,
- FIA_ATD.1 maintaining DF PIN as authenticated user security attribute,
- FIA_AFL.1(Iteration 1,2,3,4) handling unsuccessfull DF PIN authentication,
- FIA_USB.1 binding authenticated user with DF PIN,
- FIA_UAU.1 timing of authentication with DF PIN,
- FIA_UID.1 timing of identification with DF PIN,
- FMT_SMF.1 specification of management functions,
- FMT_MSA.1 (Iteration 5), FMT_MSA.2, FMT_MSA.3 defining restrictive maximum DF PIN error counter and maximum reseting number,
- FMT_MTD.1 (Iteration 3,4) access to EFs/DFs,
- FMT_SMR.1 maintaining the authenticated user,
- FPR_UNO.1 unobservability of authenticated user (Iteration 1,2,3,4)

7.2.2.3 Authorizing User to an Operation

- FAU_SAA.1 monitoring wrong key input,
- FDP_ACC.2 and FDP_ACF.1 access control SFP enforcement on initialization and personalization key,
- FIA_ATD.1 maintaining initialization key and key error counter security attributes for card initializer, personalization key and key error counter security attributes for personalizer,
- FIA_AFL.1 (Iteration 1,2,3,4) handling unsuccessfull key authentication,
- FIA_USB.1 binding initializer with initialization key, personalizer with personalization key,
- FMT_SMF.1 specification of management functions,
- FMT_MSA.1 (Iteration 1,2) defining restrictive default value to personalization and initialization keys,
- FMT_MTD.1 (Iteration 1) management of initialization and personalization keys,
- FPR_UNO.1 unobservability of initializers and personalizers are implemented by this function (Iteration 1,2,3,4).

7.2.2.4 Authentication of user to TOE

- FDP_ACC.2 and FDP_ACF.1 access control SFP enforcement on initialization, personalization and activation keys,
- FIA_UAU.4 prevents use of authentication data in external authenticate are implemented by this function,
- FCS_RND.1 generation of random numbers

7.2.2.5 Authentication of TOE to user

- FPT_TDC.1 verification and authentication commands,
- FCS_RND.1 generation of random numbers

are implemented by this function.

7.2.3 Cryptographic Keys

- FAU_ARP.1 clear key buffers,
- FCS_CKM.3 cryptographic key writing and reading,
- FCS_CKM.4 cryptographic key destruction,
- FDP_ACC.2 access control SFP enforcement on DF keys,
- FDP_ETC.1 export to and FDP_ITC.1 import from outside TSF control user data,
- FIA_ATD.1 maintaining DF key security attributes for authenticated user,
- FMT_SMF.1 specification of management functions,
- FMT_MSA.1 (Iteration 5), 2 management of keys and key error counters are implemented by this function.
- FDP_RIP.1 allocation of resource for keys.

7.2.4 Secure Messaging

- FDP_ETC.1, FDP_ITC.1 import and export of user data securely, FMT_MOF.1 management of secure messaging by card initializer, card personalizer, user, authenticated user and administrator, and FCS_RND.1 generation of random numbers is implemented by this function.

7.2.5 Integrity of the Objects

- FAU_ARP.1 giving security alarms when integrity of objects is distrupted,
- FDP_ACC.2 access control SFP enforcement on EFs, DFs and NVM,
- FDP_DAU.1 validity of command data,
- FDP_SDI.2 (Iteration 1, 2) memory integrity monitoring based on EDC (Error Detection Code),
- FAU_SAA.1 monitoring integrity check,
- FDP_ITC.1 import of data,
- FPT_FLS.1 failure with secure state,
- FPT_TST.1 memory integrity is implemented by this function.

7.2.6 Access Conditions on the DFs and EFs

- FDP_ACC.2 and FDP_ACF.1 access control SFP enforcement on EFs, DFs and NVM,
- FDP_ETC.1 and FDP_ITC.1 data transmission model enforcement on EFs and DFs,
- FMT_MTD.1 (Iteration 3,4) management of access permissions of EF and DFs to administrator and authenticated user,
- FMT_SMF.1 specification of management functions,

- FMT_SMR.1 security roles of administrator, authenticated user and user,
- FDP_ETC.1 and FDP_ITC.1 import and export of user data securely are implemented by this function,
- FDP_RIP.1 allocation of PINs, PUKs, DF, EF and DF keys on NVM.

7.2.7 Function Countering Physical Attacks

- FPT_PHP.3 and FPT_EMSEC.1 resistance to external attacks to code controlling System/DF PIN/PUK is implemented by this function.
- FAU_ARP.1 resistance of chip to tamper attacks,
- FPT_PHP.3 resistance of chip to physical attacks (low/high frequency, low/high voltage, temperature, glitch and light), random number tests are implemented by this function,
- FPT_TST.1 self tests to guarantee the correct operation of TSF.

8 Statement of Compatibility between Composite Security Target and the Platform Security Target

This chapter shows that security objectives, security requirements, and security functionality in the Composite-ST and the Platform-ST are compatible.

8.1 Seperation of the Platform-TSF

SEF1: Operating State is relevant Platform-TSF, because the operating state is monitored with sensors for the operating temperature, operating suply voltage (VDD), internal voltage (VCC), clock frequency and internal clock frequency. If one of these sensors raises an alarm due to a violation in the working conditions, than the circuit enters to the defined secure state. The defined secure state causes the chip internal reset process.

In addition, some of the critical registers, which ensures the proper operation of microprocessor, are implemented as doubled, thus they check each other continously. When these doubled registers have different values because of the attacks such as laser attacks, the circuit prevents faulty operation by entering to reset state.

Thus the FPT_FLS.1, FPT_PHP.3, and FAU_ARP.1 defined in Composite-ST are covered.

SEF2: Phase Management is relevant Platform-TSF, because the TOE must use this function for internal mode control.

SEF3: **Protection Against Snooping** is relevant platform TSF.Several mechanisms protect the TOE against snooping the design or the user data during operation and even if it is out of the operation (power down). The entire design is kept in a non standart way to prevent attacks using standart analysis methods. There are topological design countermeasures such as active shield to protect critical data.: Also bus encryptiıon mechanism between smart card dedicated CPU and the cryptographic peripherals, memory units and usage of memory encryption mechanism makes the analysis complicated.) Thus FPT_PHP.3 defined in composit ST are satisfied.

SEF4: Data Encyption and data disguising is relevant platform TSF. TOE calls the encryption function of the YITAL library functions in the main procedure of TOE. The hardware implementation of the DES, the AES and the RSA algorithms are implemented to be resistant against side channel attacks. This prevents the secure data leakage.Thus FPT_EMSEC.1 defined in composit ST are covered.

SEF5: Random Number Generation is relevant platform TSF, because the TOE is equipped with a physical random number generator which generates truly random numbers. The random data can be used from the operating system software as well as from the security enforcing functions . The TOE has the capability to subject the generated numbers to the tests of monobit, poker, runs, long run and auto correlation defined in FIPS-140-2. (Random number generator is checked by calling the YITAL library functions in the main procedure of TOE.) A random number quality test is performed in TOE's start up and an activity test is performed in background during TOE's execution. If any of the tests fail, the chip sends a reset.

Thus the FPT_PHP.3, FAU_ARP.1, FCS_RND.1 defined in composit ST are covered.

SEF6: TSF Self Test is relevant security function.

The TOE run a suite of self tests during initial startup and the cases that the operating system requires to demonstrate the correct operation of active shield, security sensors, and random number generator. As any attempt to modify the sensor devices will be detected from the test, the covered security functional requirment is FPT_TST.1

SEF7: Notification of Physaical Attack is relevant Platform-TSF, bacause the entire surface of the TOE is protected with the active shield against probing and physical attacks. Attacks over the surface are detected when the shield lines are short or open circuit. The attamped to use an opened or shortened device will be detected causing the circuit to enter reset state. Thus covered security functional requirements are FPT_PHP.3, “Notification of physical attack” and FAU_ARP “Security Alarms”defined in Composit-ST.

SEF8: Cryptographic Support is relevant Platform-TSF, because the TOE is equiped with several hardware accelerators to support standart cryptographic operations. Platform has the capability of providing the cyrpographic functions of DES/DES3, AES and RSA as hardware. The keys for this cryptographic operations are provided from the Smartcard Embeded software (enviroment).

The covered security functional requirement is FCS_COP.1 defined in Composite-ST.

8.2 Platform SFR

According to the mapping in “Coverage of Security Security Functions Rationale” in chapter 7.2 in the platform-ST the, following platform-ST SFR are relevant for the composite-ST:

FPT_FLS.1, FPT_PHP.3, FCS_COP.1, FPT_TST.1, and FCS_RND.1.

Platform SFR	Composite SFR	Rationale
FPT_FLS.1	FPT_FLS.1	The requirements match they have same meaning.
FPT_PHP.3	FPT_PHP.3, FAU_ARP.1	The requirements match they have same meaning.
FCS_RNG.1	FCS_RND.1	The requirements match, they have same meaning.
FCS_COP.1 – iteration-1,2,3,4	FCS_COP.1 iteration- 1,2,3,4,6	The requirements match, they have same meaning.
FPT_TST.1	FPT_TST.1	The requirements match they have same meaning.
FDP_ITT.1 FPT_ITT.1 FDP_IFC.1	FPT_EMSEC.1	The requirments match, they have same meaning.

Table 14- Security requirements mapping table

8.3 Platform Security Objectives

Acording to the Platform-ST paragraph 6.3 the following security objectives defined in Platform-ST are relevant for Composite-ST.

Platform Objwctive	**Composite-ST Objective**	**Rationale**
O.Phys-Probing	O.PROT_INF_LEAK	The Composite-ST security objective O.PROT_INF_LEAK matches the platform-ST security objective O.Phys-Probing according to the composite-ST FPT_PHP.3
O.Phys-Manipulation	O.PROT_PHYS_TAMPER	The Composite-ST security objective O.PROT_PHYS_TAMPER matches the platform-ST security objective O.Phys-Manipulation according to the composite-ST FPT_PHP.3
O.Malfunction	OPERATE, O.PROT_INF_LEAK	The Composite-ST security objective O.PROT_INF_LEAK and O_OPERATE matches the platform-ST security objective O.Malfunction according to the composite-ST FPT_FLS.1
O.Leak-Forced	O.PROT_INF_LEAK	The Composite-ST security objective O.PROT_INF_LEAK matches the platform-ST security objective O.Leak-Forced according to the composite-ST FPT_PHP.3
O.Abuse-Func	O.PROT_INF_LEAK, OPERATE	The Composite-ST security objective O.PROT_INF_LEAK and O_OPERATE matches the platform-ST security objective O.Abuse-Func according to the composite-ST FPT_PHP.3 and FPT_FLS.1
O.RND		This is used by embedded OS according to FCS_RND.1
O.Add-Functions	O.DIS_MEMORY, O.PROT_INF_LEAK	The Composite-ST security objective DIS_MEMORY and O.PROT_INF_LEAK matches the platform-ST security objective O.Add-Functions according to the composite-ST FCS_COP.1

Table 15- Security objectives mapping table

8.4 Platform Threats

According to the platform ST, paragraph 3.1.1, the following Platform-ST are relevant for Composite-ST.

Platform Threat	**Composite-ST Threat**	**Rationale**
T.Phys-Probing	*T.PHYS_TAMPER*	The threats match

T.Phys-Manipulation	*T.MOD, T.PHYS_TAMPER*	The threats match
T.Malfunction	*T.MOD, T.DIS*	The threats match
T.Leak-Forced	*T.MOD, T.INFO_LEAKAGE*	The threats match
T.Abuse-Func	*T.MOD T.DIS*	The threats match
T.RND		The threat is not specified in composite ST, however it it used by the composite ST.

Table 16- Threats mapping table

8.5 Platform Assumption

Acording to the Platform-ST paragraph 3.2 the following assumptions defined in Platform-ST are relevant for Composite-ST

Platform Assumption	Composite Assumption	Rational
A.Process-Sec-IC		Considered for development of the embeded software
A.Plat-Appl		Considered for development of the embeded software
A.Resp-Appl		Considered for development of the embeded software
A.Key-Function		Considered for development of the embeded software

Table 17- Assumptions mapping table

8.6 Platform-OSP

Platform OSP	Composite OSP	Rational
P. Process-TOE		Considered for development of the embeded software
P. Add-Functions		Considered for development of the embeded software
P. Key- Installation		Considered for development of the embeded software

Table 18 - OSP mapping table

9 Assurance Measures

Assurance Components	Assurance Measures (AKİS Document)	Rationale
ASE.CCL.1 ASE_ECD.1 ASE_INT.1 ASE_OBJ.2 ASE_REQ.2 ASE_SPD.1 ASE_TSS.1	UKIS_SecurityTarget	The assurance measure describes ST introduction, Conformance Claims, Security Objectives, Security Requirements and TOE Summary Specification.
ALC_CMC.4 ALC_CMS.4	UKIS_KonfigürasyonYönetimPlanı	The assurance measure describes the automated means by which only authorized changes are made to the TOE implementation and addresses the requirements for automatic generation of the TOE and automated tools used in the CM system. TOE releases are adequately identified with the version number. All Configuration Items (CI's) that comprise the TOE are under Configuration Management and are included on a CI List. The CM system is effective at ensuring that only authorized changes are made to CI's. The CM system generates records that will demonstrate that the CM system is used and include an acceptance plan.
	UKIS_SorunDurum RaporDokumani	The assurance measure addresses the documentation required to be under configuration control and describes the problem tracking system.
ALC_DEL.1 AGD_PRE.1	UKIS_TeslimveIsletim	The assurance measure addresses the requirement for secure delivery of the TOE. Secure delivery refers to tamper-evident delivery and detection of modification. Also this assurance measure addresses the requirement for installation procedures that are adequate to ensure that the user starts the TOE into a secure configuration.

ADV_FSP.4	UKIS_FonksiyonelBelirtim	The assurance measure addresses the requirement for an informal functional specification. A detailed description of the external interfaces and rationale that the TSF is completely represented is provided.
ADV_IMP.1	Source Code	The assurance measure addresses the requirements for providing the implementation representation for the TSFs.
ADV_TDS.3	UKIS_AyrıntılıTasarımDokümanı	The assurance measure addresses the requirement for TOE design documentation that describes the TOE in terms of subsystems and modules, including purpose of each module and subsystem, interrelationship between the modules and subsystems, interrelationship between modules and subsystems interfaces and the security functionality provided by each module and subsystem.
ADV_ARC.1	UKIS_SecurityArchitecture	The assurance measure addresses the requirement for secure TOE architecture.
AGD_OPE.1	UKIS_YöneticiKullaniciKilavuzu	The assurance measure addresses the requirement for administration guidance that is adequate to provide administrators with the required knowledge to securely configure and maintain the TOE within the environment. Also the assurance measure addresses the requirement for user guidance that is adequate to provide users with the required knowledge to securely access the TOE within the environment.
ALC_DVS.1 ALC_TAT.1	UKIS_GeliştirmeOrtamGüvenlik_GeliştirmeAletleri	The assurance measure addresses the requirement for site development security procedures. Also this assurance measure addresses the requirements for definition of development tools and configuration used for the TOE.
ALC_LCD.1	UKIS_KullanimOmru	This assurance addresses the requirements for life-cycle model used in the development and maintenance of

		the TOE.
ATE_COV.2 ATE_DPT.1 ATE_FUN.1	UKIS_SistemTestDokumani	The assurance measure addresses the requirement for analysis that demonstrates that the TOE was tested to the TOE design documentation and Functional Specification Documentation. Also this assurance measure includes functional tests scenarios and results.
ATE_IND.2	-	This assurance component is related with evaluater.
AVA_VAN.5	-	This assurance component is related with evaluater.

Table 19. Assurance Measures

10 REFERENCES

Common Criteria

[1] Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model; CCMB-2009-07-001, Version 3.1, Revision 3, July 2009

[2] Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Components; CCMB-2009-07-002, Version 3.1, Revision 3, July 2009

[3] Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Requirements; CCMB-2009-07-003, Version 3.1, Revision 3, July 2009

[4] Common Methodology for Information Technology Security Evaluation, Evaluation Methodology; CCMB-2009-07-004, Version 3.1, Revision 3, July 2009

[5] Smartcard Embedded Software Protection Profile, PP-9810, 19th November 1998, Common Criteria for IT Security Evaluation Protection Profile

[6] Security IC Platform Protection Profile, Version 1.0, 15.06.2007 Registered and Certified by Bundesamt für Sicherheit in der Informationstechnik (BSI) under the reference BSI-PP-0035

[7] Machine Readable Travel Document with „ICAO Application”, Basic Access Control; BSI-CC-PP-0055, Version 1.10, 25th March 2009, Bundesamt für Sicherheit in der Informationstechnik