TÜBİTAK BİLGEM

TÜBİTAK BİLGEM
ULUSAL ELEKTRONİK & KRİPTOLOJİ ARAŞTIRMA ENSTİTÜSÜ

AKIS PROJE GRUBU

AKILLI KART İŞLETİM SİSTEMİ (AKİS)
SECURITY TARGET LITE

AKİS V1.2.2N

Revision No	05
Revision Date	19.04.2011
Document Code	AKIS-ST-LITE-LITE
Developer	TUBITAK-BILGEM
Department	AKIS PROJECT GROUP

Date of Revision

Revision No	**Revision Reason**	**Date of Revision**
01	First Version	15.09.2010
02	Updated (GR24'e gore)	11.11.2010
03	Updated	31.12.2010
04	Updated	24.01.2011
05	Updated (V1.2.1 to V1.2.2'ye)	19.04.2011

CONTENT

LIST OF FIGURES

LIST OF TABLES

Table 1 Abbreviations & Glossary

AKİS	**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**
MF	**Master File**
ST	**Security Target**
TOE	**Target of Evaluation**
TPDU	**Transmission Protocol Data Unit**
TSF	**TOE Security Functionality**
DAC	**Discretionary Access Control.**

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.

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 may be in any 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 (Non-Volatile Memory) 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**.**

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.

1 SECURITY TARGET INTRODUCTION

1.1 ST Reference

ST Title: Akıllı Kart İşletim Sistemi v1.2.2n (AKİS v1.2.2n) Security Target Lite, rev 5, April 19, 2011

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

1.1.1 Operation Notation for Functional Requirements

There are four types of operations that can be applied on functional requirements. These are;

Selection: Shown by cornered brackets, bold and italicized text.

Assignment: Shown by cornered brackets and bold text.

Refinement: Beginning with "refinement:", indicated by cornered brackets and bold text.

Iteration: Indicated by assigning a number at the functional component level.

1.2 TOE Reference

TOE Identification: AKİS (Akıllı Kart İşletim Sistemi) version 1.2.2n

1.3 TOE Overview

TOE is a smart card operating system which can be used in personal identification, digital sign, health care system, smart logon, secure email. TOE:

- Is loaded into ROM of the NXP's Smart Card (P5CC080) during the manufacturing phase. P5CC080 has EAL 5+ (ALC_DVS.2, AVA.MSU.3, AVA_VLA.4) certificate.
- Does not allow loading of executable files,Communicates with the PC via card reader according to ISO/IEC 7816-4 T = 1 protocol,
- Implements user and interface authentication,
- 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),
- Follows the life cycles (activation, manufacturing, initialization, personalization, administration, operation and death) and operates functions according to the present life cycle,
- Encrypts, decrypts, digitally signs and verifies with RSA/DES/3DES cryptographic algorithms by using HW modules of the P5CC080,
- Calculates SHA-1 hash.

1.4 TOE Description

AKiS v1.2.2n Algorithms and crypto specifications are;

Authentication;
External Authenticate: DES/DES3/RSA (1024 bit)
Internal Authenticate: DES/DES3
Encryption;
DES-ECB: Plain data can be encrypted with a DES key.
DES3-ECB: Plain data can be encrypted with a DES3 (DDES) key (A-B-A key structure).
RSA2048: Plain data can be encrypted with an RSA2048 key.
Decryption;
DES-ECB: Encrypted data can be decrypted with a DES key.
DES3-ECB: Encrypted data can be decrypted with a DES3 (DDES) key (A-B-A key structure).
RSA2048: Encrypted data can be decrypted with an RSA2048 key.
Digital Sign;
RSA2048: Plain data can be signed with an RSA2048 key.
Digital Sign Verification;
RSA2048: Signed data with the length equal to the RSA2048 key modulus length can be verified with an RSA2048 key.
Data Integrity;
DES-MAC: Cryptographic checksum is calculated with a DES key.
Hash;
SHA-1: Data can be hashed with SHA-1 algorithm.

1.4.1 Smart Card Overview

Smart cards are used as electronic authentication keys, digital signs, GSM cards and bank cards. Also, they are used as electronic passports and e-government cards such as personal identification and health care cards.

Basically smart card 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. This chip consists of microprocessor, ROM, RAM, EEPROM and some hardware units (decoders, advanced crypto engine, RNG)
- Operating system (written in ROM and enables the operation of card functions using hardware units)

From the 3 parts listed above, only the third one is developed by TÜBİTAK-UEKAE. The first part is developed by a card manufacturer company (who provides the conditions that are presented in AKİS_TeslimveIsletim document) and the second part is developed by NXP Company. The second part has EAL 5+ (compatible with BSI0002) certificate. TOE operates on NXP's P5CC080 chip. Chip consists of; 8051 based microprocessor, ROM, EEPROM, RAM, Advanced Crypto Engine (ACE), Random Number Generator, MMU, UART and Timers.

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

1.4.2 TOE Components

TOE components are (Figure 4)

- 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 format which is mentioned above. Incoming TPDU packet is analysed and block type decision is made by the communication handler. TPDU can include 3 different types of block, named R, S and I 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 EEPROM. Memory manager is used to open new file, close file, delete page and attach new page.

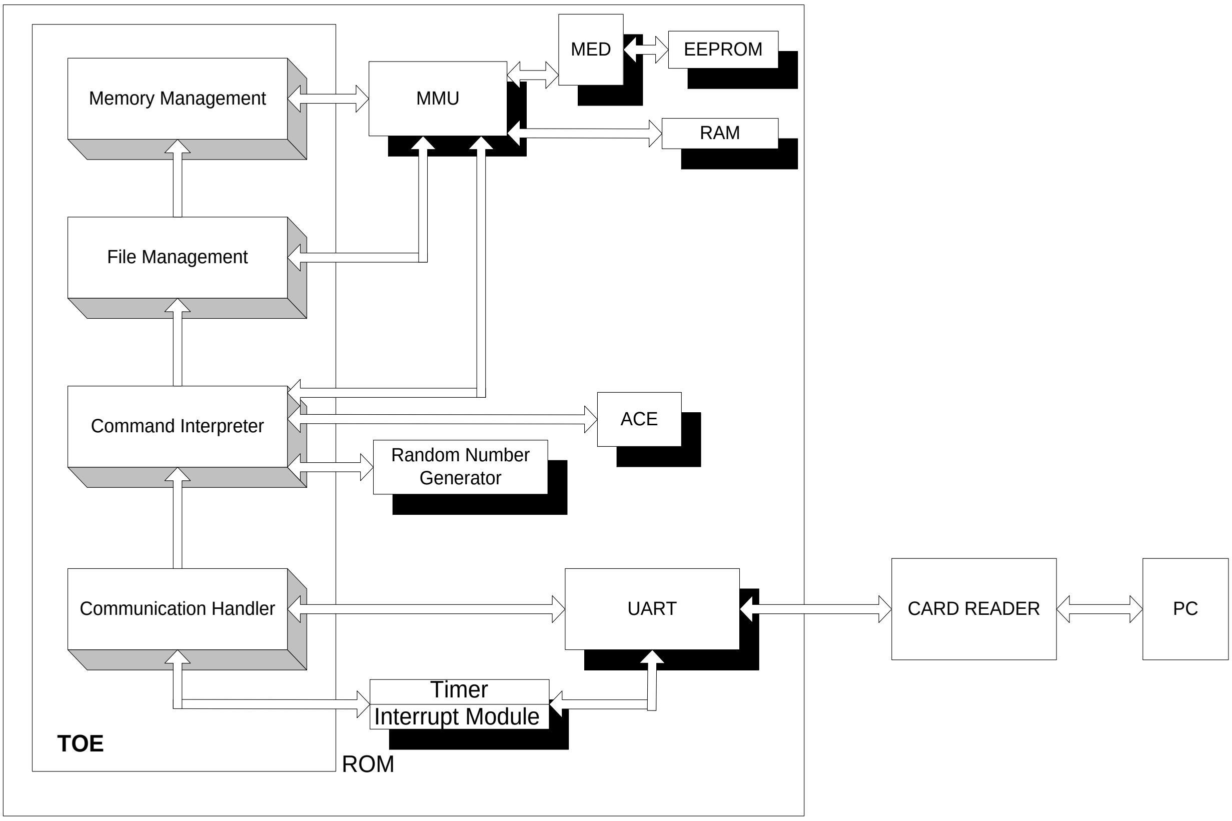


Figure 1.TOE's components and environment

1.4.3 TOE's Scope and Boundaries and Phases

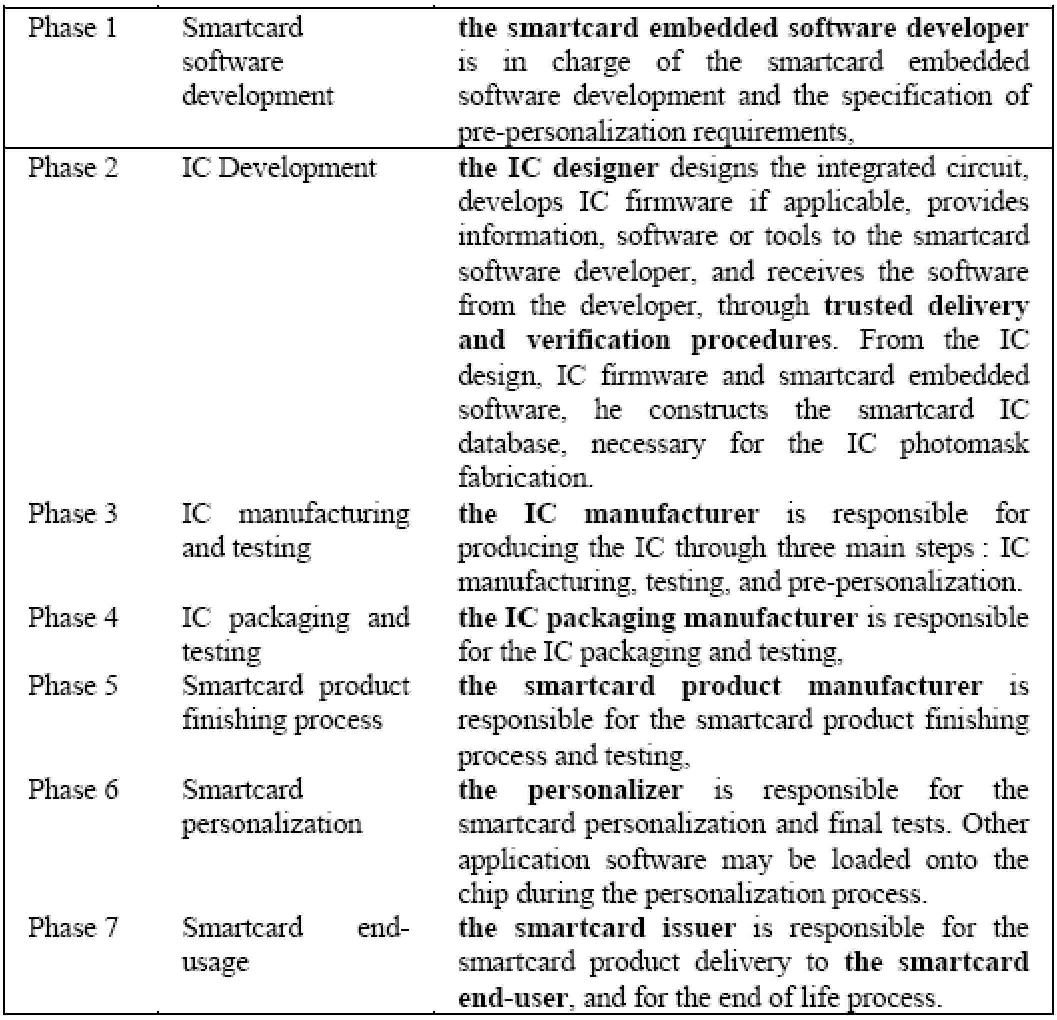
1.4.3.1 Scope and Boundaries

TOE's scope and boundaries are shown in Table 2. During TOE evaluation a PC/SC compatible smart card reader is needed. Smart card reader's driver and smart card communication software must be installed to the computer for operation.

Table 2 TOE's Scope and Boundaries

TOE	AKİS(Akıllı Kart İşletim Sistemi) v1.2.2n
Hardware	NXP P5CC080 chip (SECURE_MX51 CPU, 200K ROM , 80K EEPROM, 6K External RAM, MMU, UART, Timers, Cryptographic hardware co-processors, RNG (Random Number Generator)

TOE life cycle phases are:



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

2 CONFORMANCE CLAIM

2.1 CC Conformance Claim

- Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model, Version 3.1, Revision 3, July 2009 .
- Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Components, Version 3.1, Revision 3, July 2009 conformant.
- Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Components,Version 3.1, Revision 3, July 2009 conformant.

2.2 PP Claim

No PP Claim.

2.3 Package Claim

EAL 4 augmented (ALC_DVS.2, 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.

The augmented component ALC_DVS.2 is the high level for ALC_DVS. The augmented component AVA_VAN.5 is the high level for AVA_VAN.

3 SECURITY PROBLEM DEFINITION

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 6.1, and the TOE Security Assurance Requirements specified in Section 6.2.

The assets to be protected are:

- TOE system design,
- The basic software (including operating system program and documentation),
- TOE hex code,
- Activation key,
- The application data of the TOE (such as initialization, personalization requirements, keys, PIN/PUK).

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

3.1 Threats

The TOE as defined in Chapter 1 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)

3.1.1 Threats on all smartcard product life cycle phases (1 to 7), Figure 6

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: Application data
Phase 7: Application 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: Application data
Phase 7: Application 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: Application data
Phase 7: Application data

3.1.2 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). TOE system design, basic software, TOE hex code, activation key are subject to threats.
T.FLAW Introduction of flaws in the TOE 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, evalOS). 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. Application data and activation key 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.

3.1.3 Threats on delivery of software and related information from smartcard product life cycle phases 1, 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 developer 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.
- personalization data send by the personalizer to the smart card issuer, phase 6 to 7.

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

3.1.4 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.
T.DESIGN_IC Poor IC design leading to IC security mechanisms not meeting state of the art level. Application data is subject to threat.

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

3.1.6 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. Application data is subject to threat.

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

Table 3 Threats during phases

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/II	Class I/II	Class I/II	Class I/II
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	
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.DESIGN_IC		Class II					
T.MOD	Class II	Class II	Class I/II	Class I/II	Class I/II	Class I/II	Class I/II
T.MOD_INFO	Class II						
T.MOD_DEL	Class II	Class II	Class II			Class II	

3.2 Organizational Security Policies

OSP.SECURE_DF Creation of DFs with the secure messaging attribute.

3.3 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,(Smart card product life cycle phase 2)
   - assumptions on smartcard product life cycle phases 3 to 6.
   - assumptions on smartcard product life cycle phase 7.

2. The intended usage of the TOE.

3.3.1 Security aspects of the environment in which the TOE is intended to be used

3.3.1.1 Assumptions on the TOE delivery process from phase 1 to phase 7

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).

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.

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

There are two types of assumptions: the assumptions on the development of the TOE and the assumptions on the personnel aspects.

Secure development:

A.IC_PRODUCT the Smartcard integrated circuit is designed and built using state of art technology with the aim of achieving security objectives.

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.

3.3.1.3 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).

3.3.1.4 Assumption on smartcard product life cycle phase 7 and on delivery to these phases

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

3.3.2 Assumption on the intended usage of the TOE, related with TOE Life Cycle Phases (Figure 5) 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 knows 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 (phases 3 to 7).
O.TAMPER The TOE must prevent tampering with its security functions (phases 3 to 7).
O.FUNCTION The TOE must provide protection against unauthorized use of its software application functions (phases 3 to 7).
O.CLON The TOE functionality needs to be protected from cloning (phases 3 to 7).
O.OPERATE The TOE must ensure the continued correct operation of its security functions (phases 3 to 7).
O.DIS_MECHANISM The TOE shall ensure that the software security mechanisms are protected against unauthorized disclosure (phases 3 to 7).
O.DIS_MEMORY The TOE shall ensure that the embedded software does not allow unauthorized access to information stored in memories (phases 3 to 7).
O.MOD_MEMORY The TOE shall ensure that the embedded software does not allow unauthorized modification or corruption of the information stored in memories (phases 3 to 7).

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 Application 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.
O.FLAW The TOE must not contain flaws in design, data values or implementation.

4.2.2 Objective on smartcard product life cycle phase 2

O.MECH_IC The IC shall be designed using state of art technology focusing on:

- preventing physical tempering with its security critical parts,

- protection from cloning,
- ensuring correct operation of its security functions,
- not containing flaws in design, implementation or operation,
- protecting stored memory from unauthorized disclosure,
- protection of sensitive stored information against any corruption or unauthorized modification.

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

O.DIS_DEV The IC designer, manufacturer, 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.

4.2.4 Objectives on smartcard product life cycle phase 2 to 7

O.PRODUCT_DEV The smart card embedded software developer, 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.

4.2.5 Objective on smart card life cycle phase 6

O.SECURE_DF The personalizer must create DFs with the secure messaging attribute on.

4.2.6 Objective on the intended usage of the TOE, related with TOE Life Cycle Phases (Ref : ST Figure 5) 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 Objectived 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 3, 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. IC and embedded OS is delivered from IC manufacturer to IC packaging manufacturer (phase 4). 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. IC and embedded OS is delivered from IC manufacturer to IC packaging manufacturer (phase 4). 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.DESIGN_IC The Integrated Circuit is designed during phase 2, so the threat concerns only this phase.
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, 4 to 7, the TOE counters the threats on embedded software and data. Environment also encounters this threat from phase 1 to phase 7. 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. IC and embedded OS is delivered from IC manufacturer to IC packaging manufacturer (phase 4). 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.

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, and T.MOD
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 during smart card product life cycle phases 3 to 7.
O.TAMPER addresses illegal modification of the TOE once it is completed, thus it counters the threat T.MOD 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, and that the operations cannot be corrupted, T.MOD, during smart card product life cycle phases 3 to 7.
O.FLAW addresses the threat T.FLAW during the conception of the TOE.
O.DIS_MECHANISM addresses the Threat T.DIS. As knowledge of the security mechanisms is necessary for cloning, it also contributes to counter T.CLON. It helps to counter T.MOD by keeping confidential the security mechanisms which have to be broken to realize the threat. 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, threat T.DIS. As cloning requires knowledge of memory content. 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. 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.

Table 4 Mapping of TOE objectives to threat

Threats/T.Obj.	INTEGRITY	TAMPER	FUNCTION	OPERATE	FLAW	DIS_ MECHANISM	DIS_ MEMORY	MOD_ MEMORY	CLON
CLON			X			X	X		X
DIS				X	X	X	X		
T.PRODUCT			X						X
MOD	X	X		X	X	X		X	
FLAW					X				

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

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.

O.MECH_ACS Restricting access to the security mechanisms 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.

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, threat 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.

Table 5 Mapping of security objectives for the environment to threats relative to phase 1

Threats/E.Obj	SOFT_ACS	MECH_ACS	TI_ACS	INIT_ACS	TOOLS_ACS	SAMPLE_ACS
CLON	X	X	X	X	X	X
DIS	X	X		X		X
MOD	X	X		X		
DIS_INFO			X			
DIS_TEST			X			
T_TOOLS					X	

T_SAMPLE						X
MOD_INFO			X			

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 phase 2

O_MECH_IC addresses specifically T_DESIGN_IC.

4.3.1.3.3 Smart card product life cycle phases 2 to 6.

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, T.DESIGN_IC
- Threats on delivery of data to phases 2, 3, 4 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.

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

Threats/ E.Obj	DIS_DEV	SOFT_DLV	MECH_IC
CLON	X	X	
DIS	X	X	
DIS_DEL		X	
DIS_TEST	X		
DESIGN_IC			X
T_DEL		X	
MOD	X	X	
MOD_DEL		X	

It is demonstrated that all class II threats during phases 2 and threats concerning delivery to phases 2, 3, 4 and 6 are addressed by at least one security objectives for the environment.

4.3.1.3.4 Smart card product life cycle phases 1 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, and T.DIS_TEST

O.PRODUCT_DEV contributes to the protection of TOE data and related information including test information during phases 1 to 7, and thus addresses T.DIS, T.MOD and T.DIS_TEST. O.PRODUCT_DEV addresses directly T.T_PRODUCT and thus helps to counter T.CLON.

Table 7 Mapping of security objectives for the environment to threats relative to phases 1 to 7

Threats/ E.Obj	PRODUCT_DEV
CLON	X
DIS	X
T_PRODUCT	X
DIS_TEST	X
MOD	X

It is demonstrated that all class II threats during phases 1 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.
This section describes why the security assumptions are suitable to provide each of the security objectıves.

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.

Table 8 Mapping of security assumptions and objectives for the environment

Assumptions/E.Obj	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

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_MECH_IC is linked to A.IC_PRODUCT
O_OPERATION is linked to A.USE_OPR

Table 9 Mapping of security assumptions and objectives for the environment (Table 8 continued)

Assumptions/E.Obj	PRODUCT_DEV	MECH_ IC	OPERATION
DLV_CONTROL	X		
DLV_PROTECT	X		
DLV_AUDIT	X		
DLV_CONF	X		
DLV_RESP	X		
DLV_TRACE	X		
DLV_TRANS	X		
IC_ORG	X		
IC_PRODUCT		X	
USE_OPR			X
USE_PROD	X		
USE_SYS	X		

4.3.3 Security Objectived Related with Organizational Security Policies

This section demonstrates that the organizational security policies are suitably satisfied by the identified security objectives for the environment.

O.SECURE_DF is linked to OSP.SECURE_DF. This OSP is necessary to prevent attacks like Man in the middle.

Table 10 Mapping of OSP and objectives for the environment

OSP/E.Obj	O.SECURE_DF
OSP.SECURE_DF	X

5 EXTENDED COMPONENTS DEFINITION

There are no components identified which are not present in CC part 2 or CC part 3.

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 rev3, July 2009

6.1.1 Security Audit

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 unsuccessfull authentication attempts.**
2. **Force the card to go to activation life cycle if the life cycle data is corrupted.**
3. **Force the card to go to a reset if a tamper attack is detected.**
4. **Disable DF keys if DF keys are corrupted.**
5. **Give an error to the user if an uncontrolled write operation to EEPROM's special areas is detected.**
6. **Give a warning to the user if any corruption occurs in DFs, EFs, DF PIN, DF PUK, System PIN and System PUK.]**

FAU_SAA Security Audit Analysis

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 enforcement of the SFRs.

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

a) Accumulation or combination of [

- **A wrong input of activation key while system keys are being loaded.**
- **A wrong input of the current system keys values while system keys (ba-ka) are being changed.**
- **A wrong input of initialization key while the card's EEPROM is being erased.**
- **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.**
- **A wrong input of cryptogram while the external interface is being authenticated by the card.**
- **A wrong input of DF/System PIN during verify command**
- **A wrong input of DF/System PUK during reset retry counter command]**

known to indicate a potential security violation;

b) **[None]**

6.1.2 Cryptographic support

FCS_CKM Cryptographic Key Management

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]**.

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]**.

FCS_COP Cryptographic Operation

FCS_COP.1 Cryptographic operation Iteration 1

FCS_COP.1.1 The TSF shall perform **[digital signature/verification]** in accordance with a specified cryptographic algorithm **[RSA]** and cryptographic key sizes **[up to 2048 bits for modulus]** that meet the following: **[PKCS #1 (RSA Cryptography Standard)]**.

FCS_COP.1 Cryptographic operation Iteration 2

FCS_COP.1.1 The TSF shall perform **[encryption/decryption]** in accordance with a specified cryptographic algorithm [**RSA]** and cryptographic key sizes **[up to 2048 bits for modulus]** that meet the following: **[PKCS #1 (RSA Cryptography Standard)]**.

FCS_COP.1 Cryptographic operation Iteration 3

FCS_COP.1.1 The TSF shall perform **[encryption/decryption]** in accordance with a specified cryptographic algorithm [**DES]** and cryptographic key sizes **[8 bytes]** that meet the following: **[DES Cryptography Standard]**.

FCS_COP.1 Cryptographic operation Iteration 4

FCS_COP.1.1 The TSF shall perform **[encryption/decryption]** in accordance with a specified cryptographic algorithm **[3DES]** and cryptographic key sizes [**16 bytes]** that meet the following: [**3DES Cryptography Standard]**.

FCS_COP.1 Cryptographic operation Iteration 5

CS_COP.1.1 The TSF shall perform [**cryptographic checksum calculation/ verification]** in accordance with a specified cryptographic algorithm [**3DES/DES]** and cryptographic key sizes [**16/8 bytes]** that meet the following: [**MAC Standard]**.

FCS_COP.1 Cryptographic operation Iteration 6

CS_COP.1.1 The TSF shall perform [**external/internal authenticate]** in accordance with a specified cryptographic algorithm [**3DES]** and cryptographic key sizes [**16 bytes]** that meet the following: [**3DES Cryptography Standard]**.

FCS_COP.1 Cryptographic operation Iteration 7

CS_COP.1.1 The TSF shall perform [**external/internal authenticate]** in accordance with a specified cryptographic algorithm [**DES]** and cryptographic key sizes [**8 bytes]** that meet the following: [**DES Cryptography Standard]**.

FCS_COP.1 Cryptographic operation Iteration 8

CS_COP.1.1 The TSF shall perform [**external authenticate]** in accordance with a specified cryptographic algorithm [**RSA]** and cryptographic key sizes [**1024 bits for modulus]** that meet the following: [**PKCS #1 (RSA Cryptography Standard)]**.

6.1.3 User Data Protection

FDP_ACC Access Control Policy

FDP_ACC.2 Complete access control

FDP_ACC.2.1 The TSF shall enforce the [**AKİS 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.

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

FDP_ACF Access Control Functions

FDP_ACF.1 Security attribute based access control

FDP_ACF.1.1 The TSF shall enforce the [**AKİS access control SFP]** to objects based on the following: [**activation key, initialization key, personalization key, error counters, DF PINs, System PIN, DF error counter PUKs, System PUK, resetting and error counters, DF keys, life cycle, and access permissions of DFs and EFs].**

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.**
- **Card initializer formats the card with the initialization key.**
- **Card initializer initializes the card with the initialization key.**

- **Card initializer changes the initialization key.**
- **Card personalizer personalizes the card with the personalization key.**
- **Card personalizer changes the personalization key.**
- **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 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 authorise 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 never uses system/DF PIN or system/DF PUK information.**
- **User or authenticated user never changes life cycle.]**

FDP_DAU Data Authentication

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 **[command data]**.
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.

FDP_ETC Export from the TOE

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

FDP_ITC Import from Outside of the TOE

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 TOE.
FDP_ITC.1.2 The TSF shall ignore any security attributes associated with the user data when imported from outside the TOE.
FDP_ITC.1.3 The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: **[**

User Data Transmission Policy

- **User writes keys to DFs created without PIN.**
- **User creates EFs under DFs created without PIN.**
- **User writes EFs under MF created without PIN.**
- **User creates EF under MF created without PIN.**
- **User writes EFs under MF created without PIN.**
- **User writes EFs created with WRITE_WITHOUT_PIN access right.**

Authorized User Data Transmission Policy

- **Authorized user writes keys to DFs created with PIN.**
- **Authorized user writes to EFs under DFs created with PIN (if EFs aren't created with READ_WITHOUT_PIN or WRITE_WITHOUT_PIN).**
- **Authenticated user unblocks the DF by resetting DF PIN counter by DF PUK.**

Administrator Data Transmission Policy

- **Administrator writes keys to all DFs.**
- **Administrator writes to any EF in the card.**
- **Administrator changes any DFs/EFs access rights.**
- **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.]**

FDP_RIP Residual Information protection

FDP_RIP.1 Subset residual information protection

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

FDP_SDI Stored data integrity

FDP_SDI.2 Stored data integrity monitoring and action

FDP_SDI.2.1 The TSF shall monitor user data stored in containers controlled by the TSF for **[memory corruption]** on all objects, based on the following attributes: [**EDC (Error Detection Code).]**

FDP_SDI.2.2 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, 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.]**

6.1.4 Identification and Authentication

FIA_AFL Authentication Failures

FIA_AFL.1 Authentication failure handling Iteration 1

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 PIN with the verify command**].
FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been **[*met*]**, the TSF shall **[forbid any access to the related DF**]**.**

FIA_AFL.1 Authentication failure handling 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 When the defined number of unsuccessful authentication attempts has been **[*met*]**, the TSF shall **[force the card into death life cycle].**

FIA_AFL.1 Authentication failure handling Iteration 3

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

FIA_AFL.1 Authentication failure handling Iteration 4

FIA_AFL.1.1 The TSF shall detect when **[10]** unsuccessful authentication attempts occur related to **[changing of the system keys with Change Key command, erasing of EEPROM with Erase Files command, writing configuration data with Put Data command and authentication of the external interface with External Authenticate command].**

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

FIA_ATD User Attribute Definition

FIA_ATD.1 User attribute definition

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

Table 11 User – Security Attributes

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**

]

FIA_UAU User Authentication

FIA_UAU.1 Timing of authentication

FIA_UAU.1.1 The TSF shall allow **[execution of Select, ReadDirectoryContent, 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.

FIA_UAU.3 Unforgeable authentication

FIA_UAU.3.1 The TSF shall [***prevent***] use of authentication data that has been forged by any user of the TSF.

FIA_UAU.3.2 The TSF shall [***prevent***] use of authentication data that has been copied from any other user of the TSF.

FIA_UAU.4 Single-use Authentication Mechanisms

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

FIA_UID User identification

FIA_UID.1 Timing of identification

FIA_UID.1.1 The TSF shall allow **[execution of Select, ReadDirectoryContent, 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.

FIA_USB User-subject Binding

FIA_USB.1 User-subject binding

FIA_USB.1.1 The TSF shall associate the following 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 reseting and error counter, System/DF PUK, System/DF PUK error counter, DF keys and life cycle]**

FIA_USB.1.2 The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users: **[**

- **Card activator initializes the initialization key.**
- **Initialization key must be 16 Bytes.**
- **Card activator initializes the personalization key.**
- **Personalization key must be 16 Bytes.**
- **Card initializer initializes System/DF PIN reseting error counter.**
- **System/DF PIN resetting error counter must be in range 1 to 254.**
- **Card initializer initializes System/DF PUK reseting error counter. System/DF PUK resetting error counter must be in range 1 to 254.**
- **Administrator initializes System PIN/PUK.**
- **System PIN/PUK must be minimum 4 maximum 16 Bytes.**
- **Administrator initializes DF PIN/PUK.**
- **DF PIN/PUK must be minimum 4 maximum 16 Bytes.**
- **Authenticated user writes DF keys to DFs created with PIN.**
- **User writes DF keys to DFs created without PIN.]**

FIA_USB.1.3 The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: **[**

- **Card initializer changes the initialization key.**
- **Initialization key must be 16 Bytes.**
- **Card personalizer changes the personalization key.**
- **Personalization key must be 16 Bytes.**
- **Administrator changes System PIN/PUK.**
- **System PIN/PUK must be minimum 4 maximum 16 Bytes.**
- **Administrator changes DF PIN/PUK.**
- **DF PIN/PUK must be minimum 4 maximum 16 Bytes.**
- **Administrator changes life cycle.**
- **Authenticated user initializes DF PIN/PUK.]**

6.1.5 Security management

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.]**

FMT_MOF Management of Functions in TSF

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.]**

FMT_MSA Management of security attributes

FMT_MSA.1 Management of Security Attributes Iteration 1

FMT_MSA.1.1 The TSF shall enforce the **[AKİS access control SFP]** to restrict the ability to **[*modify*]** the security attributes **[initialization key]** to **[card initializer].**

FMT_MSA.1 Management of Security Attributes Iteration 2

FMT_MSA.1.1 The TSF shall enforce the **[AKİS access control SFP]** to restrict the ability to **[*modify*]** the security attributes **[personalization key]** to **[card personalizer].**

FMT_MSA.1 Management of Security Attributes Iteration 3

FMT_MSA.1.1 The TSF shall enforce the **[AKİS access control SFP]** to restrict the ability to **[*modify*]** the security attributes **[PUK, life cycle]** to **[administrator].**

FMT_MSA.1 Management of Security Attributes Iteration 4

FMT_MSA.1.1 The TSF shall enforce the **[AKİS access control SFP]** to restrict the ability to **[*modify*]** the security attributes **[Secure messaging and PIN]** to **[authenticated user].**

FMT_MSA.1 Management of Security Attributes Iteration 5

FMT_MSA.1.1 The TSF shall enforce the **[AKİS access control SFP]** to restrict the ability to **[write *and delete*]** the security attributes **[DF keys]** to **[authenticated user].**

FMT_MSA.1 Management of Security Attributes Iteration 6

FMT_MSA.1.1 The TSF shall enforce the [**AKİS access control SFP]** to restrict the ability to [**write *and delete*]** the security attributes [**DF keys]** to **[administrator].**

FMT_MSA.2 Secure security attributes

FMT_MSA.2.1 The TSF shall ensure that only secure values are accepted for **[System PIN, System PUK, DF PIN, DF PUK, DF keys]**

FMT_MSA.3 Static attribute initialization

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

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.

FMT_MTD Management of TSF data

FMT_MTD.1 Management of TSF data Iteration 1

FMT_MTD.1.1 The TSF shall restrict the ability to **[initialize]** the **[initialization key and personalization key]** to **[card activator].**

FMT_MTD.1 Management of TSF data Iteration 2

FMT_MTD.1.1 The TSF shall restrict the ability to **[*modify*]** the **[System/DF PIN, System/DF PIN error counter]** to **[administrator].**

FMT_MTD.1 Management of TSF data Iteration 3

FMT_MTD.1.1 The TSF shall restrict the ability to **[read/write]** the **[EFs and DFs]** to **[authenticated user.]**

FMT_MTD.1 Management of TSF data Iteration 4

FMT_MTD.1.1 The TSF shall restrict the ability to **[create/read/write/delete]** the **[EFs and DFs]** to **[administrator].**

FMT_MTD.1 Management of TSF data Iteration 5

FMT_MTD.1.1 The TSF shall restrict the ability to [**create/read/write/delete]** the [**EFs and DFs]** to **[administrator].**

FMT_SMR Security management roles

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.]**

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

6.1.6 Privacy

FPR_UNO Unobservability

FPR_UNO.1 Unobservability Iteration 1

FPR_UNO.1.1 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 Unobservability Iteration 2

FPR_UNO.1.1 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 Unobservability Iteration 3

FPR_UNO.1.1 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 Unobservability Iteration 4

FPR_UNO.1.1 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.]**

6.1.7 Protection of TOE security functions

FPT_FLS Fail secure

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.]**

FPT_PHP TSF Physical protection

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 SFRs are always enforced.

[

Table 12 Physical Tampering Scenarios

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 supply	**Temperature out of range**	**Go to a reset**

]

FPT_TDC Inter-TSF TSF data consistency

FPT_TDC.1 Inter-TSF basic TSF data consistency

FPT_TDC.1.1 The TSF shall provide the capability to consistently interpret **[commands]** when shared between the TSF and another trusted IT product.

FPT_TDC.1.2 The TSF shall use **[verification and authentication commands]** when interpreting the TSF data from another trusted IT product.

FPT_TST TSF self test

FPT_TST.1 TSF Testing

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

- **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 EEPROM 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]**

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 **[*TSF data*.]**

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

6.2 Security assurance requirements

This section specifies the assurance requirements for the TOE. Details of the assurance components specified in this section can be found in part 3 of the Common Criteria.

Table 12 below provides a complete listing of the Security Assurance Requirements for the TOE. These requirements consists of the Evaluation Assurance Level 4 (EAL 4) components as specified in Part 3 of the Common Criteria, augmented with ALC_DVS.2: Development security, AVA_VAN.5: Advanced methodical vulnerability analysis.

Table 13 Assurance Requirements

Assurance Class	**Component ID**	**Component Title**
Development	ADV_ARC.1	Security architecture description
	ADV_FSP.4	Complete functional specification
	ADV_IMP.1	Implementation representation of the TSF
	ADV_TDS.3	Basic modular design
Guidance documents	AGD_OPE.1	Operational user guidance
	AGD_PRE.1	Preparative procedures
Life-cycle support	ALC_CMC.4	Production support, acceptance procedures and automation
	ALC_CMS.4	Problem tracking CM coverage
	ALC_DEL.1	Delivery procedures
	ALC_DVS.2	Sufficiency of security measures
	ALC_LCD.1	Developer defined life-cycle model
	ALC_TAT.1	Well-defined development tools
Security Target evaluation	ASE_CCL.1	Conformance claims
	ASE_ECD.1	Extended components definition
	ASE_INT.1	ST introduction
	ASE_OBJ.2	Security objectives
	ASE_REQ.2	Derived security requirements
	ASE_SPD.1	Security problem definition
	ASE_TSS.1	TOE summary specification
Tests	ATE_COV.2	Analysis of coverage
	ATE_DPT.1	Testing: basic design
	ATE_FUN.1	Functional testing
	ATE_IND.2	Independent testing - sample
Vulnerability assesment	AVA_VAN.5	Advanced methodical vulnerability analysis

6.3 Security requirements rationale

This section provides the rationale for necessity and sufficiency of security requirements, demonstrating that each of the security objectives is addressed by at least one security requirement, and that every security requirement is directed toward solving at least one objective.

6.3.1 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 13) 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.

Table 14 Mapping of security functional requirements and IT objectives

SFR/T.OBJ	INTEGRITY	TAMPER	FUNCTION	OPERATE	O.FLAW	DIS_MEC HANISM	DIS_ MEMORY	MOD_ MEMORY	CLON
FAU_ARP.1	X					X	X	X	
FAU_SAA.1	X					X	X	X	
FCS_CKM.3		X					X		Partial
FCS_CKM.4		X					X		Partial
FCS_COP.1 (I.1, I.2, I.3, I.4, I.5, I.6, I.7, I.8)		X					X		Partial
FDP_ACC.2		X	X	Partial		X	X	X	Partial
FDP_ACF.1		X	X			X	X	X	Partial
FDP_DAU.1	X	X		Partial				X	Partial
FDP_ETC.1							X		Partial
FDP_ITC.1	X							X	
FDP_RIP.1		X					X		Partiall
FDP_SDI.2	X			Partial				X	Partial
FIA_AFL.1 (I.1, I.2, I.3, I.4)		X		X					Partial
FIA_ATD.1		X							Partial
FIA_UAU.1		X					X	X	Partial
FIA_UAU.3		X					X	X	Partial
FIA_UAU.4							X	X	Partial
FIA_UID.1		X					X	X	Partial
FIA_USB.1		X					X	X	Partial
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, I.6)		X				X	X	X	Partial
FMT_MSA.2		X							
FMT_MSA.3		X				X	X	Partial	Partial
FMT_MTD.1 (I.1, I.2, I.3, I.4, I.5)		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					
FPT_PHP.3		X							
FPT_TDC.1	X	X							
FPT_TST.1	X								

6.3.1.1 Security Functional Requirements Sufficiency

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) support the access control to the assets. These functions cooperate to meet the security objectives of O.TAMPER, 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.TAMPER, 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, O.TAMPER, 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.TAMPER, and O.DIS_MEMORY objectives 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 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) and FIA_ATD.1 meet the security objective O.TAMPER by handling authentication failures and maintaining user's attributes. FIA_AFL.1 (I1, I2, I3, I4) also meets the security objective O.OPERATE. They both also contribute to the correct operation of TOE, O.CLON.

Identification and authentication functional requirements FIA_UAU.1 and FIA.UAU.3 meet O.TAMPER, 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 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.TAMPER, 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.TAMPER, 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.TAMPER, O.OPERATE, O.DIS_MECHANISM, O.DIS_MEMORY and O.MOD_MEMORY objectives by managing the security functions which fulfill these objectives.

Management of TSF data functional requirements FMT_MSA.1 (I1, I2, I3, I4), FMT_MSA.2 and FMT_MSA.3 meet O.TAMPER objectives. FMT_MSA.1 (I1, I2, I3, I4) and FMT_MSA.3 also 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.TAMPER, O.DIS_MECHANISM, O.DIS_MEMORY and O.MOD_MEMORY objectives by management of TSF data.

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

FPT_PHP.3 by resisting tampering meets O.TAMPER by resisting to physical attacks.

FPT_TDC.1 functional requirement meets O.INTEGRITY and O.TAMPER objectives by assuring inter TSF data consistency.

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

6.3.1.2 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.

The assurance requirements specified in this ST are precisely as defined in EAL 4 with one higher hierarchical component (ALC_DVS.2) and AVA_VAN.5. ALC_DVS.2 has no dependency.

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

The following table (Table 14) 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.

Table 15 Dependency analysis

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	FDP_ACC.2 : Complete Access Control	FDP_ACF.1
7	FDP_ACF.1 : Security attributes based Access Control	FDP_ACC.1 FMT_MSA.3
8	FDP_DAU.1 : Basic Data Authentication	No dependencies
9	FDP_ETC.1 : Export of user data without security attributes	FDP_ACC.1 or FDP_IFC.1
10	FDP_ITC.1 : Import of user data without security attributes	FDP_ACC.1 or FDP_IFC.1 FMT_MSA.3
11	FDP_RIP.1 : Subset residual information protection	No dependencies
12	FDP_SDI.2 : Stored data integrity monitoring and action	No dependencies
13	FIA_AFL.1 : Authentication failure handling.	FIA_UAU.1
14	FIA_ATD.1 : User attribute definition	No dependencies
15	FIA_UAU.1 : Timing of authentication	FIA_UID.1
16	FIA_UAU.3 : Unforgeable authentication	No dependencies
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 : Secure security attributes	FDP_ACC.1 or FDP_IFC.1 FMT_MSA.1 FMT_SMR.1
23	FMT_MSA.3 : Static attribute 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 attack	No dependencies
30	FPT_TDC.1 : Inter-TSF basic TSF data consistency	No dependencies
31	FPT_TST.1 : TSF testing	No dependencies

Dependency Rationale:

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 EEPROM size is limited and many of these events may bring the card to an insecure state where recording itself could open a security breach.

6.3.2 Security assurance requirements rationale

6.3.2.1 Assurance Measures Rationale

Table 15 demonstrates correspondence rationale between assurance components and assurance measures.

Table 16 Assurance Measures Rationale

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	AKIS V1.2.2n Security Target	The assurance measure describes ST introduction, Conformance Claims, Security Objectives, Security Requirements and TOE Summary Specification.
ALC_CMC.4 ALC_CMS.4	AKISV1.2.2n_Konfigürasyon _Yönetim_Planı	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.
	AKISV1.2.2n_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	AKISV1.2.2n_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	AKISV1.2.2n_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	AKISV1.2.2n_Ayrintili Tasarim Dokumani	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	AKİSV1.2.2n_Guvenlik Mimari Dokumani	The assurance measure addresses the requirement for secure TOE architecture.
AGD_OPE.1	* AKISV1.2.2n_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.2	AKISV1.2.2n_GelistirmeOrtam	The assurance measure addresses the requirement for site development

ALC_TAT.1	Guvenlik_GelistirmeAletleri	security procedures. Also this assurance measure addresses the requirements for definition of development tools and configuration used for the TOE.
ALC_LCD.1	AKISV1.2.2n_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	AKISV1.2.2n_Sistem Test Dokumani	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.

6.3.2.2 Security Assurance Requirements meet Security Objectives for the environment 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 16) demonstrates which Assurance component contribute to the satisfaction of each security objective for the environment. For clarity, the table does not identify indirect dependencies.

Table 17 Mapping of assurance components and security objectives for the environment during the development phase.

Assurance Componenets /E.Obj	SOFT_ACS	MECH_ ACS	TI_ACS	INIT_ACS	TOOLS_ ACS	SAMPLE_ACS
ALC_DVS.2	X	X	X	X	X	X

Assurance component ALC_DVS.2 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.

6.3.2.3 TOE Life Cycle Phases (Ref: ST Figure 5) except activation phase

The assurance measure related with AGD_OPE.1 meets O.Operation environmental objective. Assurance measure canalizes the users and administrators when TOE gives a warning message related with corrupted objects.

6.4 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 ALC_DVS.2) 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.

6.4.1 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.

6.4.1.1 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.3 and **FIA_UAU.4** prevent 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.

6.4.1.2 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.3 and **FIA_UAU.4** prevent 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.

6.4.1.3 Deactivation

Prevention of deactivation is derived as described below:

The access control SFP detailed in **FDP_ACF.1** 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.

6.4.1.4 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 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 2048 algorithm can be used for this operation, so the referenced key must be an RSA 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 2048 algorithm can be used for this operation, so the referenced key must be an RSA 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 2048, 3DES (DDES) and DES 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 requirements:

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

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 2048, DES3 and DES 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 EEPROM 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 EEPROM 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 RSA, DES3 and DES).

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

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

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 goes to the death life cycle. Only the administrator can change the System PIN. System PIN must be minimum 4, maximum 16 digits.

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), 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. During PIN change operation, if the operation is interrupted by taking out the card from the card reader, old PIN is valid.

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.

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 18).

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 4), FMT_MSA.2, 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 RSA2048 and DES3 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, Change Key, Erase Files and External Authenticate (activation).

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 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 and DES 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:

- *FDP_ACC.2, FDP_ACF.1,*
- *FIA_UAU.3, 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 and DES 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*

7.1.3 Cryptographic Keys

Proprietary key access function is used to write and erase application keys from EEPROM, RAM/XRAM. Each key has unique ID number per application. Two components of DES and 3DES keys are written with in the same APDU command whereas each component of RSA keys is written in a seperate command with different parameters by TOE.

While the keys are written into TOE, the algorithm of the key and type of the cryptographic operations will be used with this key are determined by APDU command. The key is not allowed to be written if algorithm of key is inconsistent with determined cryptographic operations for this key.

DES and 3DES keys can not be used with sign and verify signature operations. They can be used with Ext. Auth., Int. Auth., Encryption, Decryption, MAC and verify MAC operations.

RSA keys can not be used with Ext. Auth., Int. Auth., MAC, verify MAC operations. They can be used with Encryption, Decryption, verify signature operations.

All key lenghts are checked whether the length of the key is meaningful or not. The key is not allowed to be written with an invalid length.

Content of all system keys and DF keys are checked if they are not entirely composed of 0xFF. All keys must include at least one byte different from 0xFF. Otherwise, the key is not allowed to be written.

Proprietary key access function reads the modulus and public components of RSA keys which are loaded to the application. Since these components are public, they can be read without any authentication.

DES, 3DES keys and PDAT, QDAT, DPDAT, DQDAT, QINVDAT components of RSA keys are not given outside the card. An error response is produced if these components are tried to be read.

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 4), 2, FMT_SMF.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 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 3DES 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 requirements :

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

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.

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.

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

- *FAU_ARP.1,*
- *FDP_ACC.2, FDP_DAU.1, FDP_SDI.2, FDP_RIP.1,*
- *FPT_FLS.1, FPT_TST.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 18).

Table 18 Access conditions for TOE EFs/DFs

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

- 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*

7.1.7 Function Countering Physical Attacks

7.1.7.1 Countering System/DF PINs and System/DF PUK Attacks

In order to prevent unexpected jumps in critical code points which may be caused by external attacks, there is a double check in code lines controlling System/DF PIN and System/DF PUK.

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

- *FPT_PHP.3*

7.1.7.2 Physical Sensors

P5CC080 chip produces an NMI when code, data and IRAM areas are attacked. In this chip, there are different sensors for the physical attacks such as low/high frequency, low/high voltage, temperature, glitch and light detectors. Chip produces a HW reset signal or NMI interrupt when these sensors sense an abnormal situation. TOE goes to a reset (soft RESET) state if NMI is produced.

Random number generator and Sanity of the Physical sensors are checked by calling the NXP library functions in the main procedure of TOE.

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

- *FAU_ARP.1*
- *FPT_PHP.3*

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).

7.2.1.4 Decryption

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

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 2) handling unsuccessfull System PINauthentication,
- FIA_USB.1 binding administrator withSystem PIN, defining restrictive maximum System PIN error counter and maximum reseting number,
- FMT_SMF.1 specification of management functions,

- FMT_MSA.1 (Iteration 3),2,3 defining default values of system PIN error counter and maximum resetting number; restricting secure values for PIN/PUK; restrict modification on PUK and life cycle,
- FMT_MTD.1 (Iteration 2) change System/DF PIN,
- FMT_SMR.1 maintaining the administrator role,
- FPR_UNO.1 unobservability of administrator is implemented by this function. (Iteration 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) handling unsuccessfull DF PIN authentication,
- FIA_USB.1 binding authenticated user with DF PIN, defining restrictive maximum DF PIN error counter and maximum reseting number,
- 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 4), defining default values of DF PIN error counter and maximum resetting number; restricting secure values for DF PIN/PUK,
- FMT_MTD.1 (Iteration 3) access to EFs/DFs,
- FMT_SMR.1 maintaining the authenticated user,
- FPR_UNO.1 unobservability of authenticated user, (Iteration 3)

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 3,4)handling unsuccessfull system keys authentication,
- FIA_USB.1 defining restrictive default value to personalization and initialization keys,
- FMT_SMF.1 specification of management functions,
- FMT_MSA.1 (Iteration 1,2) binding initializer with initialization key, personalizer with personalization key,
- FMT_MTD.1 (Iteration 1) management of initialization and personalization key,

- FPR_UNO.1 (Iteration 1,2) unobservability of initializers and personalizers are implemented by this function.

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.3 and FIA_UAU.4 prevent use of authentication data in external authenticate are implemented by this function.

7.2.2.5 Authentication of TOE to user

FPT_TDC.1 verification and authentication commands 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 are implemented by this function.
- FIA_AFL.1 (Iteration 4) key error counters are implemented by this function.

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 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 EEPROM,
- FDP_DAU.1 validity of command data,
- FDP_SDI.2 memory integrity monitoring based on EDC (Error Detection Code),
- FDP_RIP.1 deallocation DF PINs, DF, EF and DF keys on EEPROM,
- 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 EEPROM,
- 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

7.2.7 Function Countering Physical Attacks

7.2.7.1 Countering System/DF PIN and System/DF PUK Attacks

FPT_PHP.3 resistance to external attacks to code controlling System/DF PIN/PUK is implemented by this function.

7.2.7.2 Physical Sensors

- 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) is implemented by this function.