Document Administration

Recipient

Department	Name
SecEDocRnD / PAD	**Secure e-Documents Research and Development / PADERBORN**

For the attention of

Department	Name

Summary

The following document comprises the Security Target Lite for a TOE evaluated according to Common Criteria Version 3.1 Revision 3. The TOE being subject of the evaluation is the smartcard product

MICARDO V4.0 R1.0 eHC V1.2

from Morpho Cards GmbH. The IT product under consideration shall be evaluated according to CC EAL 4 augmented with AVA_VAN.5.

Keywords

Target of Evaluation (TOE), Common Criteria, IC, Dedicated Software, Smartcard Embedded Software, Basic Software, Application Software, Security Objectives, Assumptions, Threats, TOE Security Function (TSF), TOE Security Enforcing Function (SEF), Level of Assurance, Strength of Functions (SOF), Security Functional Requirement (SFR), Security Assurance Requirement (SAR), Security Function Policy (SFP)

Responsibility for updating the document

SecEDocRnD / PAD Secure e-Documents RnD / PADERBORN

Morpho Cards GmbH

MICARDO V4.0 R1.0 eHC V1.2

Security Target Lite

Document Id:	3MIC3EVAL.CSL.0019
Archive:	3
Product/project/subject:	MIC3EVAL (Micardo V3 Evaluierung)
Category of document:	CSL (Security Target Lite)
Consecutive number:	0019
Version:	V1.00
Date:	17 April 2014
Author:	Karsten Klohs
Confidentiality:	

Checked report:

Authorized (Date/Signature):

Accepted (Date/Signature):

OdsDok V2.00

Document Organisation

i Notation

None of the notations used in this document need extra explanation.

ii Official Documents and Standards

See Bibliography.

iii Revision History

Version	Type of change	Author / team
V1.00	Final Version of ST-Lite based on the ST	Thomas Hoffmann

Table of Contents

1 ST Introduction

1.1 ST Identification

This Security Target refers to the smartcard product “MICARDO V4.0 R1.0 eHC V1.2” (TOE) provided by Morpho Cards GmbH for a Common Criteria evaluation.

Title:	Security Target Lite - MICARDO V4.0 R1.0 eHC V1.2
Document Category:	Security Target for a CC Evaluation
Document ID:	Refer to Document Administration
Version:	Refer to Document Administration
Publisher:	Morpho Cards GmbH
Confidentiality:	
TOE:	“MICARDO V4.0 R1.0 eHC V1.2” (Smartcard Product containing IC with Smartcard Embedded Software, including eHC Application, intended to be used within the German Health Care System)
Certification ID:	BSI-DSZ-CC-0861
IT Evaluation Scheme:	German CC Evaluation Scheme
Evaluation Body:	SRC Security Research & Consulting GmbH
Certification Body:	Bundesamt für Sicherheit in der Informationstechnik (BSI)

This Security Target has been built in conformance with Common Criteria V3.1 Revision 3.

1.2 ST Overview

Target of Evaluation (TOE) and subject of this Security Target (ST) is the smartcard product “MICARDO V4.0 R1.0 eHC V1.2” developed by Morpho Cards GmbH.

The TOE is realised as Smartcard Integrated Circuit (IC with contacts) with Smartcard Embedded Software, consisting of the

MICARDO V4.0 Operating System platform

and the dedicated

electronic Health Card Application (eHC Application)

as intended to be used for the German Health Care System.

The TOE`s eHC Application is based on the MICARDO V4.0 Operating System platform.

In particular, the TOE´s platform and its technical functionality and inherently integrated security features are designed and developed under consideration of the following specifications, standards and requirements:

- Functional and security requirements defined in the specification /EXS_EHC_1/ and /EXS_EHC_2/ for the electronic Health Card (eHC) as employed within the German Health Care System
- Requirements from the Protection Profile /BSI_PP_EHC/
- Technical requirements defined in ISO 7816, Parts 1, 2, 3, 4, 8, 9, 15

The TOE is intended to be used as electronic Health Card (eHC) within the German Health Care System. More detailed:

The eHC Application running on the underlying MICARDO V4.0 Operating System platform is implemented according to the requirements in /EXS_EHC_1/. The eHC Application in the sense of this ST covers all elementary files at the MF-level, the DF.HCA, the DF.ESIGN, the DF.CIA.ESIGN as defined in /EXS_EHC_2/ and further Morpho specific files.

Under technical view, the TOE comprises the following components:

- Integrated Circuit (IC) with Crypto Library "NXP SmartMX2 P60C080PVC(y) Secure Smart Card Controller with Cryptographic Library V1.0 as IC Dedicated Support Software" provided by NXP Semiconductors GmbH
- Smartcard Embedded Software comprising the MICARDO V4.0 Operating System platform (designed as native implementation) and the dedicated eHC Application for the German Health Care System provided by Morpho Cards GmbH

Remark: The NXP P60C080PVC(y) is a yield improved variant of the P60C080PVC with no impact on security. The change has no effect on assurance which is proved and documented by the BSI. For more information see the "Assurance Continuity Maintenance Report" /NXP_IC_MA/. The P60 dedicated Crypto Library V1.0 is certified in the dutch scheme under certification number NSCIB-CC-12-36243, /ST-IC+CL/.The configuration of the TOE as eHC will be done by Morpho Cards GmbH prior to the delivery of the product. In any case, the TOE contains the dedicated eHC Application.

The TOE contains at its delivery unalterable identification information on the delivered configuration. Furthermore, the TOE provides authenticity information which allows an authenticity proof of the product.

The TOE will be delivered from Morpho Cards GmbH in the following variants:

- Delivery as not-initialised module or smartcard:
  The delivery of the modules resp. smartcards will be combined with the delivery of the customer specific initialisation file (in particular containing the evaluated eHC Application) developed by Morpho Cards GmbH. The initialisation file is sent (by a secured transfer way) to the Initialiser for loading the EEPROM initialisation data into the TOE during its initialisation phase whereat the production requirements defined in the Guidance for the Initialiser (as well delivered by Morpho Cards GmbH) have to be considered.
  In the case of the delivery of modules, the last part of the smartcard finishing process, i.e. the embedding of the delivered modules and final (card) tests, is task of the customer.

- Delivery as initialised module or smartcard:
  The initialisation of the modules resp. smartcards will be performed by Morpho Cards GmbH prior to the delivery of the TOE to the customer.
  In the case of the delivery of modules, the last part of the smartcard finishing process, i.e. the embedding of the delivered modules and final (card) tests, is task of the customer.

The form of the delivery of the TOE does not influence the security features of the TOE in any way. However, in the case of the delivery of the product in initialised form, the initialisation process at Morpho Cards GmbH will be considered in the framework of the TOE´s CC evaluation. As the manner of the delivery of the TOE does not affect the security of the TOE in any way the TOE will be named in the following with “eHC” for short, independently of its form of delivery.

In order to be compliant with the requirements from the German Health Care System the TOE will be evaluated according to CC EAL 4 augmented with AVA_VAN.5.

The CC evaluation and certification of the TOE implies the proof for compliance of the TOE´s eHC Application with the corresponding specifications /EXS_EHC_1/ and /EXS_EHC_2/ and their requirements.

The main objectives of this ST are

- to describe the TOE as a smartcard product intended to be used as eHC
- to define the limits of the TOE
- to describe the assumptions, threats and security objectives for the TOE
- to describe the security requirements for the TOE
- to define the TOE security functions

1.3 TOE Overview

The TOE Overview refers to the Chapter 1.2 of the Protection Profile /BSI_PP_EHC/.
The Chapter 1.2.1, 1.2.2, 1.2.3 and 1.2.4 can be mapped to the Chapters 1.3.1, 1.3.2, 1.3.3 and 1.3.4 of this document.

1.3.1 Usage of the TOE

The security target addresses the security services provided by this card, mainly:

- Mutual Authentication between the eHC and a Health Professional Card (HPC) or a Security Module Card (SMC),
- Mutual Authentication between the eHC and a security device (e.g. for online update of contract data in the card),
- Authentication of the cardholder by use of one of two PINs, called PIN.CH and PIN.home (which of these PINs is relevant depends on the service the cardholder wants to use),
  Note: Both of these PINs are used for general functions of the eHC. The electronic signature application (see below) requires a separate third PIN for its exclusive purposes.
- Secure storage of contractual and medical data, with respect to confidentiality, integrity and authenticity of these data,

- Authentication of the card using a private key and a X.509 certificate and
- Document content key decipherment using a private key.

Note: The eHC may contain an electronic signature application for the cardholder. This application is subject to the requirements for electronic signatures as defined in national and European law. This TOE does not include an electronic signature application.

1.3.2 Major security features of the TOE

German health insurance companies issue electronic Health Cards to patients insured by them. The card is used by the cardholders, when they use health care services, which are covered by the insurance. A picture of the patient is printed on the card in order to support identification. The eHC contains data for

- cardholder identification,
- contractual and financial information to be exchanged between cardholder and health care provider and/or the health insurance company and
- medical data.

In detail the functionality of the card is defined in the specifications /EXS_EHC_1/ and /EXS_EHC_2/

The following list gives an overview of the main security services provided by the electronic Health Card during the usage phase. In order to refer to these services later on, short identifiers are defined.

Service_Asym_Mut_Auth_w/o_SM[1]:
Mutual Authentication using asymmetric techniques between the eHC and a Health Professional Card (HPC) or a Security Module Card (SMC) without establishment of a Secure Channel.

This service is meant for situations, where the eHC requires authentication by a HPC or SMC, but where the following data exchange is done without help of a security module.

Service_Asym_Mut_Auth_with_SM:
Mutual Authentication using asymmetric techniques between the eHC and a Security Module Card (SMC) or another security module with establishment of a Secure Channel.

This service is meant for situations, where the eHC requires authentication by a SMC or another security module, which provides similar functionality, and where the following data exchange is done with the help of this security module and can therefore be encrypted and/or secured by a MAC.

Service_Sym_Mut_Auth_with_SM:
Mutual Authentication using symmetric techniques between the eHC and a security module with establishment of a Secure Channel.

[1]The Abbreviation SM here stands for Secure Messaging, which is the card security protocol realising a secure channel.

This service is meant for situations, where the eHC communicates with a central security module, which shares symmetric keys with the card. This may be a security module of the health insurance organisation, when managing the patient contractual data, or a module of the Download Service Provider, which may add new applications to the eHC (or manage the existing ones).

Service_User_Auth_PIN:

The cardholder authenticates himself with one of his PINs, either PIN.CH or PIN.home.

This service is meant as a support service for some of the other services, which may require user authentication. In addition it provides privacy protection because certain data in the card (or secured by the card) can only be accessed after user authentication. In particular this applies to sensitive medical data.

Functions to change the PIN or to unblock the PIN, when it was blocked (because of successive false PIN entries) are supporting this service. For the latter the PIN unblocking code (PUC) is used, this authentication will be called **Service_User_Auth_PUC**.

Service_Privacy:

The cardholder may deactivate sensitive medical data in the eHC. In order to use this service he authenticates himself with a PIN.

This service allows the cardholder to prevent health care providers from accessing data, which the cardholder doesn't want them to know. Note, that that the name Service_Privacy doesn't mean that this is the only privacy related service. In fact all other services also support privacy.

Service_Client_Server_Auth:

The eHC implements a PKI application, which in particular allows using the TOE as an authentication token for an authentication of a client to a server (by means of an asymmetric method using X.509 certificates). The eHC contains two different keys and corresponding certificates for this service. In order to use this service the cardholder authenticates himself with a PIN. One of the keys can also be used without authentication by the cardholder, but requires authentication by a HPC or SMC in this case.

This service may for example be useful if the cardholder wants to access a server provided by the health insurance organisation, where confidential data of the cardholder are managed. So it can also be seen as an additional privacy feature.

Note, that a potential authentication of the server to the client is not supported by the eHC.

Service_Data_Decryption:

The eHC implements a PKI application, which in particular allows using the TOE as a data decryption token. Symmetric document encipherment keys, which are themselves encrypted with the cards public key can only be decrypted with the help of the card. There are two sets of asymmetric key pairs in the eHC to allow the following two possibilities of authentication for this service:

- In order to use this service the cardholder authenticates himself with a PIN.
- One of the keys can also be used without authentication by the cardholder, but requires authentication by a HPC or SMC in this case.

This service is meant for situations, where confidential data are stored on a server, but shall only be accessible with the cardholder's permission or with the authentication of a health professional. So it can also be seen as a privacy feature.

Service_Card_Management:
The eHC allows creation of new applications and management of existing applications to the card management system. This is secured by the service Service_Sym_Mut_Auth_with_SM.

Service_Logging:
The eHC provides a file, which allows to store information about the fifty last accesses to medical data in the card. The card itself doesn't control the content of these data; it is up to the authorised persons, who have write access to these data, to write them correctly.

Note: The eHC may implement a PKI application, which in particular makes it possible to use the TOE as an electronic signature creation device for qualified signatures. The specification of requirements for this service is **not** covered by this PP/BSI_PP_EHC/.. Annex 7.1 gives information on the combination of this PP with PPs suitable for electronic signature services.

In detail the functionality of the card is defined in the specifications /EXS_EHC_1/ and /EXS_EHC_2/

1.3.3 TOE Type

The Target of Evaluation (TOE) is a smart card, the electronic Health Card (eHC), which is conformant to the specification documents /EXS_EHC_1/ and /EXS_EHC_2/. The size of the card is type ID-1 according to ISO 7810 (the usual credit-card-size).
The card is a card with contacts according to ISO 7816-1 to –3. If it has an additional contactless interface, none of the eHC functions shall be accessible via this interface.

1.3.4 Required non-TOE hardware/software/firmware

The TOE is the electronic Health Card (contact based smart card). For the usage of this smart card an appropriate terminal resp. the health care system is necessary.

1.4 TOE Description

1.4.1 TOE definition

The TOE definition refers to the Chapter 1.3.1 of the Protection Profile /BSI_PP_EHC/. The overall system including the TOE and its environment are intended to comply to the relevant German legal regulations, in particular the "Gesetz zur Modernisierung der Gesetzlichen Krankenversicherung" (GKV-Modernisierungsgesetz – GMG), the "Sozialgesetzbuch" (SGB) and the privacy legislation ("Datenschutzgesetze des Bundes und der Länder").

The TOE comprises the following parts

TOE_IC, consisting of:

- the circuitry of the eHC's chip (the integrated circuit, IC) and

- the IC Dedicated Software with the parts IC Dedicated Test Software and IC Dedicated Support Software

TOE_ES,

- the IC Embedded Software (operating system)

TOE_APP,

- the eHC applications (data structures and their content)

and

guidance documentation delivered together with the TOE.

Note: The short terms TOE_IC, TOE_ES and TOE_APP will be used were appropriate in the rest of this document in order to refer to these parts of the TOE.

The following table contains an overview of all deliverables associated to the TOE:

TOE component	Description / Additional Information	Type	Transfer Form
TOE-IC	NXP SmartMX2 P60C080PVC(y) Secure Smart Card Controller (incl. its IC Dedicated Software, covering in particular the Crypto Library)	HW / SW	Delivery of not-initialised / initialised modules or smart-cards Delivery of initialisation files in electronic form (if applicable)
TOE-ES	Smartcard Embedded Software / Part Basic Software (implemented in ROM/EEPROM of the microcontroller)	SW	
TOE-APP	Smartcard Embedded Software / Part Application Software (containing the eHC Application implemented in the EEPROM of the microcontroller)	SW	
Note: The TOE will be delivered from Morpho Cards GmbH as not-initialised or initialised product (module / smartcard). To finalize the TOE as not-initialised product, the initialisation file developed by Morpho Cards GmbH must be loaded during the initialisation phase by the Initialiser (Morpho Cards GmbH or other initialisation facility).			
User Guide for User of the MICARDO platform	User guidance for the User of the MICARDO Operating System platform	DOC	Document in paper / electronic form
User Guide for Initialiser of the eHC Card	User guidance for the Initialiser of the eHC Card	DOC	Document in paper / electronic form
User Guide for Personaliser of the eHC Card	User guidance for the Personaliser of the eHC Card (in particular the eHC Application)	DOC	Document in paper / electronic form
Identification Data Sheet of the eHC Card	Data Sheet with information on the actual identification data and configuration of the eHC Card delivered to the customer	DOC	Document in paper / electronic form
Aut-Key of the eHC Card	Public part of the authentication key pair relevant for the authenticity of the eHC Card Note: The card´s authentication key pair is generated by Morpho Cards GmbH and depends on the TOE´s configuration delivered to the	KEY	Document in paper form / electronic file

TOE component	Description / Additional Information	Type	Transfer Form
	customer. Furthermore, the key pair may be chosen customer specific.		
Pers-Key of the eHC Card	Personalisation key relevant for the personalisation of the eHC Card Note: The card´s personalisation key pair is generated by Morpho Cards GmbH and depends on the TOE´s configuration delivered to the customer. Furthermore, the key may be chosen customer specific.	KEY	Document in paper form / electronic file

Note: Deliverables in paper form require a personal passing on or a procedure of at least the same security. For deliverables in electronic form integrity and authenticity attribute will be attached.

1.4.2 Integrated Circuit (IC) with its Dedicated Software

Basis for the TOE´s Smartcard Embedded Software is the "NXP SmartMX2 P60C080PVC(y) Secure Smart Card Controller with Cryptographic Library V1.0 as IC Dedicated Support Software". The microcontroller and its Dedicated Software are developed and produced by NXP Semiconductors GmbH (within phase 2 and 3 of the smartcard product life-cycle, see Chapter 1.5).

Detailed information on the IC Hardware, the IC Dedicated Software (in particular the Crypto Library) and the IC interfaces can be found in /ST_IC/ and /ST-IC+CL/.

1.4.3 Smartcard Embedded Software

The Smartcard Embedded Software of the TOE comprises the MICARDO V4.0 Operating System platform and applications running on this platform and is therefore divided into two parts with specific contents:

- TOE-ES: Basic Software (MICARDO V4.0 Operating System platform)
- TOE-APP: Application Software (Application Layer with dedicated eHC Application)

Each part of the Smartcard Embedded Software is designed and developed by Morpho Cards GmbH in phase 1 of the smartcard product life-cycle (see Chapter 1.5). Embedding of the Smartcard Embedded Software into the TOE is performed in the later phases 3 and 5.

The main parts of the Basic Software are brought into the card by the IC manufacturer in form of the ROM mask and stored in the User-ROM of the IC (phase 3). The Application Software, and perhaps additional parts of the Basic Software, are located in the EEPROM area and are later on loaded by specific initialisation routines of the TOE (phase 5). Hereby, the loading requires an encrypted and with a cryptographic checksum secured initialisation file. The necessary keys for securing the initialisation process are stored inside the IC during production time.

1.4.3.1 TOE_ES: Basic Software

The Basic Software of the Smartcard Embedded Software comprises the MICARDO V4.0 Operating System platform of the TOE. Its main and security related parts are stored in the User-ROM of the underlying IC and are brought into the smartcard in form of the so-called ROM mask during the production process of the IC within phase 3 of the smartcard product life-cycle (see Chapter 1.5).

The MICARDO V4.0 Operating System platform of the TOE is designed as proprietary software consisting of two layers. In detail, the integral parts of the TOE´s operating system consist of the MICARDO Layer and the Initialisation Module. Both are based on a Native Platform which serves as an abstraction layer towards the IC. On the other side, the MICARDO Layer and the Initialisation Module provide an interface between the operating system and the overlying Application Layer with the dedicated eHC Application.

The MICARDO Layer implements the executable code for the card commands and all general technical and security functionality of the MICARDO V4.0 Operating System platform as data objects and structures, file and object handling, security environments, security resp. cryptographic algorithms, key and PIN management, security states, access rules, secure messaging etc.

As mentioned, the Native Platform of the TOE´s operating system serves as an abstraction layer between the MICARDO Layer resp. the Initialisation Module and the IC. For this task, it provides essential operating system components and low level routines concerning memory management, I/O handling, transaction facilities, system management, security features and cryptographic mechanisms.

For the cryptographic features, the Native Platform makes use of a specific module, the Crypto Library, which supports and implements the TOE´s core cryptographic functionality. The Crypto Library is provided as IC Dedicated Support Software by the underlying IC. In view of the Smartcard Embedded Software, the Crypto Library is accessible only via the Native Platform.

For the initialisation process of the TOE conducted within phase 5 of the smartcard product life-cycle (see Chapter 1.5) the operating system of the TOE puts dedicated initialisation routines at disposal which are solely accessible during the initialisation phase and which are realised within the Initialisation Module. After the initialisation has been successfully completed these commands are no longer available. Furthermore, the functionality of deleting the complete initialisation file after the initialisation (deletion of the whole EEPROM area) is disabled for the TOE.

The Initialisation Module puts the following features at disposal:

- specific initialisation routines

- specific test routines for the EEPROM area

Loading of an initialisation file is only possible by use of the TOE´s specific initialisation routines. Hereby, the initialisation file to be loaded has to be secured before with an encryption and a cryptographic checksum, both done with dedicated keys of the TOE.

The test routines for the EEPROM area can be used for a check of the correct functioning of the memory.

Furthermore, the Initialisation Module manages the specific states of the TOE´s operating system according to specified and unalterable rules.

In order to support the personalisation process the MICARDO V4.0 Operating System contains a personalisation module. This module provides a dedicated set of personalisation commands. These commands are only available after successful authentication with the personalisation key and are restricted to modify data intended for personalisation. Furthermore, the personalisation modules allows for the establishment of a trusted channel to secure the transfer of confidential data to the card. The personalisation commands are permanently disabled after successful personalisation.

1.4.3.2 TOE_APP: Application Software

The Application Software part of the TOE´s Smartcard Embedded Software comprises the Application Layer and is directly set-up on the TOE´s Basic Software. It consists of the TOE´s dedicated eHC Application which is implemented according to the requirements in /EXS_EHC_1/ and /EXS_EHC_2/.

The Application Software will be brought into the smartcard in cryptographically secured form during the initialisation process within phase 5 of the smartcard product life-cycle (see Chapter 1.5). The initialisation process uses the specific initialisation routines of the TOE´s operating system, and the Application Software will be stored in the EEPROM area of the IC.

The eHC offers the capability to check its authenticity. For this purpose, the TOE contains the private part of a dedicated RSA authentication key pair over which by an internal authentication procedure the authenticity of the eHC can be proven. The authentication key pair depends on the Initialisation File (containing the Application Software to be initialised) and its configuration and may be chosen customer specific. The corresponding public part of the authentication key pair is delivered through a trusted way to the external world.

Furthermore, the TOE contains a data area for storing identification data of the TOE and its configuration. The data area will be filled in the framework of the initialisation of the TOE with a specific operating system command and can be read out with a further specific operating system command. Once the identification data have been written, there is afterwards no change possible.

1.5 eHC life cycle model

The following description is a short summary of the eHC life cycle model based on a common model normally used for smart cards. The TOE life cycle is described in terms of the seven life cycle phases as usually defined for smart cards, see for example the Smartcard IC Platform PP. They are summarized in the following table:

Phase		Description
Phase 1	**Smartcard Embedded Software Development**	The **Internal Smartcard OS Developer** is in charge of • the Smartcard OS development • the development of an initialisation file (image / init tab) • the development of initialisation software updates (only if supported by the product) • the specification of IC initialisation and pre-personalisation requirements (though the actual data for pre-personalisation come from Phase 4, 5 resp. 6). • the development of in-field software updates (only if supported by the product) The purpose of the Embedded Software designed during phase 1 is to control and protect the TOE during phases 4 to 7 (product usage).The global security requirements of the TOE are such that it is mandatory during the development phase to anticipate the security threats of the other phases. The **Internal Smartcard OS Developer** is the **Morpho Cards GmbH** which implements: - the operating system MICARDO V4.0 - the initialisation files for the product configurations - in-field updates supported by OS by LOAD APPLICATION mechanism The **External IC Dedicated Software Developer** • develops the IC Dedicated Software (e.g. a crypto library) The **External IC Dedicated Software Developers** are - NXP Semiconductors GmbH which contributes the component Crypto Library on the Smart MX2 P60C080PVC(y) The **External or Internal Application Software Developer** Morpho Cards GmbH: - develops application software intended to be loaded on the operating system platform The **External Product Configurator** customer - establishes a customer configuration of the product The **External Initialisation File Finisher(not applicable for eHC product)**: • finalises the Initialisation File / InitTab (e.g. insertion of additional verification data) • delivers the Initialisation File to the Initialiser through trusted delivery and verification procedures

Phase 2	**IC Development**	The **External IC Designer** NXP Semiconductors GmbH • designs the IC, • provides information, software or tools to the Smartcard Embedded Software Developer, and • (optionally) receives and integrates the Smartcard Embedded Software (only Basic Software) from the developer through trusted delivery and verification procedures (ROM based product). From the IC design, IC Dedicated Software and Smartcard Embedded Software, the **External IC Designer** NXP Semiconductors GmbH • constructs the smartcard IC database, necessary for the IC photomask fabrication.
Phase 3	**IC Manufacturing and Testing**	The **External IC Manufacturer** NXP Semiconductors GmbH is responsible for • producing the IC through three main steps: - IC manufacturing, - IC testing, and - IC pre-personalisation. - (optionally) receives and loads the Smartcard OS (Flash based product) - (optionally) receives and loads the Initialisation File - (optionally) installs IC dedicated software during the manufacturing process The **External IC Mask Manufacturer** NXP Semiconductors GmbH • generates the masks for the IC manufacturing based upon an output from the smartcard IC database.
Phase 4	**IC Packaging and Testing**	The **External IC Packaging Manufacturer** Morpho Cards GmbH is responsible for • the IC packaging (production of modules) and • testing.
Phase 5	**Composite Product Integration**	The **Internal Initialiser** Morpho Cards GmbH related to the description in Chapter 1.5.1 or the **External Initialiser** related to the description in Chapter 1.5.1 is responsible for • (optionally) the initialisation of the TOE (in form of initialisation of the modules of phase 4 or complete smartcards) and • its testing. The **External Smartcard Product Manufacturer or** Morpho Cards GmbH is responsible for - the embedding of the modules for the TOE and the card production Final card tests only aim at checking the quality of the card production, in particular concerning the bonding and implantation of

		the modules.
Phase 6	**Smartcard Personalisation**	The **External or Internal Personaliser** Morpho Cards GmbH are responsible for • the smartcard personalisation and • administration of the smartcard (loading or deleting of objects, files or applications on customer wish, as far as allowed by the configuration of the TOE) • final tests. The personalisation of the smartcard can include the printing of the (card holder specific) visual readable data onto the physical smartcard, and the writing of (card holder specific) TOE User Data and TSF Data into the smartcard.
Phase 7	**Smartcard End-Usage**	The **External Smartcard Issuer** or Morpho Cards GmbH(for delivery) is responsible for • the smartcard product delivery to the smartcard end-user, and the end of life process. The **External In-Field Administrator** customer is allowed to • administration of the smartcard (loading or deleting of objects, files or applications on customer wish, as far as allowed by the configuration of the TOE) • loading of software updates (only if allowed by the configuration of the TOE)

Table 1: Smart Card Life Cycle Overview

The following paragraphs describe, how the application of the CC assurance classes is related to these phases.

The CC does not prescribe any specific life cycle model. However, in order to define the application of the assurance classes, the CC assume the following implicit life cycle model consisting of three phases:

- TOE development (including the development as well as the production of the TOE)
- TOE delivery
- TOE operational use

For the evaluation of the eHC the phases 1 up to 4 as defined in Table 1 are part of the TOE development in the sense of the CC. The phases 6 and 7 are part of the operational use in the sense of the CC. The phase 5 may be part of one of these CC phases or may be split between them depending on the specific model used by the TOE Manufacturer. The core scope of the evaluation consists of phases 1 to 5.

The TOE in this security target is developed and evaluated in such a way that:

- All executable software in the TOE is covered by the evaluation.

- The data structures and the access rights to these data as defined in the eHC specification /EXS_EHC_2/ are covered by the evaluation.

1.5.1 Delivery Options for the Certified Product

The certified product can be delivered in one of the following delivery packages:

- to an external initialiser as a un-initialised smartcard or module with the dedicated initialisation files, the product data sheet, and the guidance for the initialiser and the personaliser. In this case the product is delivered after phase 4 in the life-cycle model.
- to an external personaliser as initialised smartcard or module, the product data sheet and the guidance for the personaliser. In this case the product is delivered after phase 5 of the life-cycle model.

1.5.2 Delivery Procedures Relevant for the Product

The following delivery procedures of the generic life-cycle model apply to the product which forms the TOE:

- The delivery of the *Crypto Library Components* from NXP Semiconductors GmbH to the Morpho Cards GmbH
- The delivery of the MICARDO ROM mask from the Morpho Cards GmbH to NXP Semiconductors GmbH.
- The delivery of *security ICs including pre-perso data, and the operating system and the dedicated software* from NXP Semiconductors GmbH to the Morpho Cards GmbH or another initialiser approved by gematik mbH
- The delivery of initialised smartcards from the Morpho Cards GmbH or another initialiser approved by the gematik mbH to the Morpho Card GmbH or another personaliser approved by the gematik mbH.
- The delivery of personalised smartcards from the Morpho Cards GmbH or another personaliser approved by the gematik mbH to a customer requesting the product.

1.6 TOE Intended Usage

Introducing information on the intended usage of the TOE is given within Chapter 1.2. The present chapter will provide additional and more detailed information on the Operating System platform and on the eHC Application residing on the card at delivery time point.

In general, the MICARDO V4.0 Operating System platform is designed as multifunctional platform for high security applications. Therefore, the TOE provides an Operating System platform with a wide range of technical functionality and an adequate set of inherently integrated security features.

The MICARDO V4.0 Operating System platform supports the following services:

- On-card-generation of RSA key pairs of high quality (with appropriate key lengths)
- Different signature schemes (based on RSA with appropriate key lengths and padding schemes)
- Different encryption schemes (based on DES and RSA with appropriate key lengths and padding schemes)

- Key derivation schemes
- PIN based authentication scheme
- Different key based authentication schemes (based on DES and RSA, with / without session key agreement)
- Hash value calculation
- Random number generation of high quality
- Calculation and verification of cryptographic checksums
- Verification of CV certificates
- Protection of the communication between the TOE and the external world against disclosure and manipulation (Secure Messaging)
- Protection of files and data by access control functionality
- Life-cycle state information related to the Operating System itself as well as to all objects processed by the card
- Confidentiality of cryptographic keys, PINs and further security critical data
- Integrity of cryptographic keys, PINs and further security critical data
- Confidentiality of operating system code and its internal data
- Integrity of operating system code and its internal data (self-test functionality)
- Resistance of crypto functionality against Side Channel Analysis (SPA, DPA, TA, DFA)
- Card management functionality
- Channel management (with separation of channel related objects)

To support the security of the above mentioned features of the TOE, the MICARDO V4.0 Operating System platform provides appropriate countermeasures for resistance especially against the following attacks:

- Cloning of the product
- Unauthorised disclosure of confidential data (during generation, storage and processing)
- Unauthorised manipulation of data (during generation, storage and processing)
- Identity usurpation
- Forgery of data to be processed
- Derivation of information on the private key from the related public part for on-card-generated RSA key pairs
- Side Channel Attacks

The resistance of the TOE against such attack scenarios is reached by usage of appropriate security features already integrated in the underlying IC as well as by implementing additional appropriate software countermeasures.

The specific eHC Application of the TOE comprises a file system with objects, access rules and data according to the requirements in /EXS_EHC_1/ and /EXS_EHC_2/. The eHC and its dedicated eHC Application provide the main security services listed in chapter 1.3.1.

2 Conformance Claim

This security target claims conformance to

- Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model, July 2009, version 3.1 Revision 3, CCMB-2009-07-001
- Common Criteria for Information Technology Security Evaluation, Part 2: Security functional components, July 2009, version 3.1 Revision 3, CCMB-2009-07-002
- Common Criteria for Information Technology Security Evaluation, Part 3: Security assurance components, July 2009, version 3.1 Revision 3, CCMB-2009-07-003

as follows

- Part 2 extended,
- Part 3 conformant,
- Package conformant to EAL4 augmented with AVA_VAN.5.

This PP claims strict conformance to the Common Criteria Protection Profile electronic Health Card Version 2.9, 19th April 2011 BSI-CC-PP-0020-V3-2010-MA-01.

In order to avoid redundancy and to minimize the evaluation efforts, the evaluation of the TOE will be conducted as a composite evaluation and will make use of the evaluation results of the CC evaluation of the underlying semiconductor "NXP SmartMX2 P60C080PVC(y) Secure Smart Card Controller with Cryptographic Library V1.0 as IC Dedicated Support Software" provided by NXP Semiconductors GmbH. The IC incl. its IC Dedicated Software is evaluated according to Common Criteria EAL 6 augmented with ALC_FLR.1 and ASE_TSS.2. The certification number of the IC is BSI-DSZ-CC-0837 with Assurance Maintenance Report BSI-DSZ-CC-0837-MA-01 /NXP_IC_MA/.

3 Security Problem Definition

The Security Problem Definition (SPD) is the part of a PP, which describes

- **assets**, which the TOE shall protect,
- **subjects**, who are users (human or system) of the TOE or who might be threat agents (i.e. attack the security of the assets),
- **operational security policies** , which describe overall security requirements defined by the organisation in charge of the overall system including the TOE (in particular this may include legal regulations, standards and technical specifications),
- **threats** against the assets, which shall be averted by the TOE together with its environment,
- **assumptions** on security relevant properties and behaviour of the TOE's environment.

3.1 Introduction

3.1.1 Assets

For a detailed description of the TOE´s assets refer to /BSI_PP_EHC/, Chapter 3.1.1.

3.1.2 Subjects

For a detailed description of the subjects, who can interact with the TOE refer to /BSI_PP_EHC/, Chapter 3.1.2.

3.2 Organisational Security Policies

For a detailed description of the organisational security policies related to the TOE´s dedicated eHC Application refer to /BSI_PP_EHC/, Chapter 3.2.

3.3 Threats

For a detailed description of the threats to be averted by the TOE independently or in collaboration with its IT environment please refer to document /BSI_PP_EHC/, Chapter 3.3.

3.4 Assumptions

For a detailed description of the assumptions related to the TOE refer to /BSI_PP_EHC/, Chapter 3.4.

4 Security Objectives

4.1 Security Objectives for the TOE

The security objectives for the TOE address the aspects of identified threats to be countered by the TOE and organizational security policies to be met by the TOE.

For a detailed description of the specific security objectives related to the TOE´s dedicated eHC Application refer to /BSI_PP_EHC/, Chapter 4.1.

4.2 Security Objectives for the Environment of the TOE

For a detailed description of the specific security objectives related to the environment of the TOE´s dedicated eHC Application refer to /BSI_PP_EHC/, Chapter 4.2.

4.3 Security Objectives Rationale

For a detailed description of the security objectives rationale of the TOE refer to /BSI_PP_EHC/, Chapter 4.3.

5 Extended Component Definition

This security target only uses the SFR from the part 2 of the Common Criteria and the extended components defined in /BSI_PP_EHC/, Chapter 5. These additional security functional requirements are the family FCS_RND, the family FMT_LIM and the family FPT_EMSEC.

FCS_RND describes the functional requirements for random number generation used for cryptographic purposes.

FMT_LIM defines requirements that limit the capabilities and availability of functions in a combined manner. Note that FDP_ACF restricts the access to functions whereas the Limited capability of this family requires the functions themselves to be designed in a specific manner.

FPT_EMSEC defines requirements to mitigate intelligible emanations.

6 Security Requirements

The CC allows several operations to be performed on functional requirements; *refinement*, *selection*, *assignment*, and *iteration* are defined in Part 2 of the CC. Each of these operations is used in this ST.

The **refinement** operation is used to add detail to a requirement, and thus further restricts a requirement. Refinement of security requirements is either

- denoted by the word “refinement” in bold text and the added/changed words are in bold text or
- included in text as underlined text and marked by a footnote.

In cases where words from a CC requirement were deleted, a separate attachment indicates the words that were removed.

The **selection** operation is used to select one or more options provided by the CC in stating a requirement. Selections that have been made by the PP authors are denoted as underlined text and the original text of the component is given by a footnote. Selections filled in by the ST author are *italicized*.

The **assignment** operation is used to assign a specific value to an unspecified parameter, such as the length of a password. Assignments that have been made by the PP authors are denoted by showing as underlined text and the original text of the component is given by a footnote. Assignments filled in by the ST author are *italicized*.

The **iteration** operation is used when a component is repeated with varying operations. Iteration is denoted by showing a slash “/”, and the iteration indicator after the component identifier.

6.1 Security Functional Requirements for the TOE

This section on security functional requirements (SFR) for the TOE is divided into sub-sections following the main security functionality.

6.1.1 Cryptographic support (FCS)

FCS_CKM.1/SM Cryptographic key generation – Secure Messaging Keys

Hierarchical to:	No other components.
FCS_CKM.1.1/ SM	The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm card-to-card authentication with secure messaging [2] and specified cryptographic

[2] [*assignment: cryptographic key generation algorithm*]

key size *168bit*[3] that meet the following: eHC specification /EXS_EHC_1/, Part 1 [7.2] [4]which refers to /ANSI X9.63/ Chapter 5.6.3.

Dependencies: [FCS_CKM.2 Cryptographic key distribution or FCS_COP.1 Cryptographic operation]
FCS_CKM.4 Cryptographic key destruction

FCS_CKM.4 Cryptographic key destruction

Hierarchical to: No other components.

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method *erasure of a 3TDES session key*[5] that meets the following: *physical erasure of the key*[6].

Dependencies: [FDP_ITC.1 Import of user data without security attributes or FDP_ITC.2 Import of user data with security attributes or FCS_CKM.1 Cryptographic key generation]

FCS_COP.1/Hash Cryptographic operation – Hash Algorithm

Hierarchical to: No other components.

FCS_COP.1.1/ Hash The TSF shall perform hashing [7] in accordance with a specified cryptographic algorithm *SHA-256*[8] and cryptographic key sizes none [9] that meet the following: eHC specification, Part 1 [7.1] [10]/SHA/.

Dependencies: [FDP_ITC.1 Import of user data without security attributes or FDP_ITC.2 Import of user data with security attributes or FCS_CKM.1 Cryptographic key generation]
FCS_CKM.4 Cryptographic key destruction

FCS_COP.1/CCA_SIGN Cryptographic operation – Digital Signature-Creation for Card-to-Card Authentication

Hierarchical to: No other components.

[3] [assignment: *cryptographic key sizes*]
[4] [assignment: *list of standards*]
[5] [assignment: *cryptographic key destruction method*]
[6] [assignment: *list of standards*]
[7] [assignment: *list of cryptographic operations*]
[8] [assignment: *cryptographic algorithm*]
[9] [assignment: *cryptographic key sizes*]
[10] [assignment: *list of standards*]

FCS_COP.1.1/ CCA_SIGN	The TSF shall perform digital signature-creation[11] in accordance with a specified cryptographic algorithm *RSA*[12] and cryptographic key sizes *of 2048 bit modulus length*[13] that meet the following: eHC specification /EXS_EHC_1/, Part 1 [7.6][14], with references to /ISO 9796-2/ according to the algorithm. The description of the padding variant for RSA_ISO9796_2_DS1_SIGN can be found in Chapter 8.2.2 of /ISO 9796-2/.
Dependencies:	[FDP_ITC.1 Import of user data without security attributes or FDP_ITC.2 Import of user data with security attributes or FCS_CKM.1 Cryptographic key generation] FCS_CKM.4 Cryptographic key destruction

FCS_COP.1/CCA_VERIF Cryptographic operation – Digital Signature-Verification for Card-to-Card Authentication

Hierarchical to:	No other components.
FCS_COP.1.1/ CCA_VERIF	The TSF shall perform digital signature-verification[15] in accordance with a specified cryptographic algorithm *RSA*[16] and cryptographic key sizes *of 2048 bit modulus length*[17] that meet the following: eHC specification /EXS_EHC_1/, Part 1 [7.6][18], with references /ISO 9796-2/ according to the algorithm. The description of the padding variant for RSA_ISO9796_2_DS1_VERIFY can be found in Chapter 8.3 of /ISO 9796-2/
Dependencies:	[FDP_ITC.1 Import of user data without security attributes or FDP_ITC.2 Import of user data with security attributes or FCS_CKM.1 Cryptographic key generation] FCS_CKM.4 Cryptographic key destruction

FCS_COP.1/CSA Cryptographic operation – Digital Signature-Creation for Client-Server Authentication

Hierarchical to:	No other components.

[11] [assignment: *list of cryptographic operations*]
[12] [assignment: *cryptographic algorithm*]
[13] [assignment: *cryptographic key sizes*]
[14] [assignment: *list of standards*]
[15] [assignment: *list of cryptographic operations*]
[16] [assignment: *cryptographic algorithm*]
[17] [assignment: *cryptographic key sizes*]
[18] [assignment: *list of standards*]

FCS_COP.1.1/ CSA	The TSF shall perform digital signature-creation[19] in accordance with a specified cryptographic algorithm *RSA*[20] and cryptographic key sizes *of 2048 bit modulus length*[21] that meet the following: eHC specification /EXS_EHC_1/, Part 1 [7.6][22], with references to /PKCS1/ according to the algorithm. The description of the padding variant RSASSA-PSS-SIGN can be found in Chapter 8.1.1 and 9.1.1 of /PKCS1/.
Dependencies:	[FDP_ITC.1 Import of user data without security attributes or FDP_ITC.2 Import of user data with security attributes or FCS_CKM.1 Cryptographic key generation] FCS_CKM.4 Cryptographic key destruction

FCS_COP.1/Asym_DEC Cryptographic operation – Asymmetric Decryption

Hierarchical to:	No other components.
FCS_COP.1.1/ ASYM_DEC	The TSF shall perform decryption[23] in accordance with a specified cryptographic algorithm *RSA*[24] and cryptographic key sizes *of 2048 bit modulus length*[25] that meet the following: eHC specification /EXS_EHC_1/, Part 1 [7.6][26], with references to /PKCS1/ or /ISO 9796-2/ according to the algorithm. The description of the padding variants for RSAES-PKCS1-V1_5 can be found in Chapter 7.8.2.1 with reference to Chapter 7.2.2 of /PKCS1/ and for RSA-OAEP in Chapter 7.8.2.2 with reference to Chapter 7.1.2 of /PKCS1/ .
Dependencies:	[FDP_ITC.1 Import of user data without security attributes or FDP_ITC.2 Import of user data with security attributes or FCS_CKM.1 Cryptographic key generation] FCS_CKM.4 Cryptographic key destruction

FCS_COP.1/Sym Cryptographic operation – Symmetric Encryption / Decryption

Hierarchical to:	No other components.

[19] [assignment: *list of cryptographic operations*]
[20] [assignment: *cryptographic algorithm*]
[21] [assignment: *cryptographic key sizes*]
[22] [assignment: *list of standards*]
[23] [assignment: *list of cryptographic operations*]
[24] [assignment: *cryptographic algorithm*]
[25] [assignment: *cryptographic key sizes*]
[26] [assignment: *list of standards*]

FCS_COP.1.1/ Sym: The TSF shall perform encryption and decryption[27] in accordance with a specified cryptographic algorithm *3DES in CBC mode*[28] and cryptographic key sizes *168 bit*[29] that meet the following: eHC specification /EXS_EHC_1/, Part 1 [7.3] which refers to [ANSI 3.92], [30]/FIPS 46-3/ and for CBC /EXS_NIST_SP800_38A/ Chapter 6.2.

Dependencies: [FDP_ITC.1 Import of user data without security attributes or FDP_ITC.2 Import of user data with security attributes or FCS_CKM.1 Cryptographic key generation]
FCS_CKM.4 Cryptographic key destruction

FCS_COP.1/MAC Cryptographic operation – MAC

Hierarchical to: No other components.

FCS_COP.1.1/ MAC: The TSF shall perform generation and verification of message authentication code [31] in accordance with a specified cryptographic algorithm *Retail MAC*[32] and cryptographic key size *168*[33] that meet the following: eHC specification /EXS_EHC_1/, Part 1 [7.6][34]/ANSI X9.19/.

Dependencies: [FDP_ITC.1 Import of user data without security attributes or FDP_ITC.2 Import of user data with security attributes or FCS_CKM.1 Cryptographic key generation]
FCS_CKM.4 Cryptographic key destruction

FCS_RND.1 Quality metric for random numbers

Hierarchical to: No other components.

FCS_RND.1.1: The TSF shall provide a mechanism to generate random numbers that meet *deterministic RNG of quality class DRG.3*[35].

Dependencies: No dependencies.

Application note: The detailed description of DRG.3 is equal to the used functionality out of the crypto library /ST-IC+CL/ which is conformant to the specification in /AIS_2031_RNG/. For this product only the 3DES mode is used and the AES mode should not be considered and is left out in the following statements.

(DRG.3.1) If initialized with a random seed using a PTRNG of class PTG.2 as random

[27] [assignment: *list of cryptographic operations*]
[28] [assignment: *cryptographic algorithm*]
[29] [assignment: *cryptographic key sizes*]
[30] [assignment: *list of standards*]
[31] [assignment: *list of cryptographic operations*]
[32] [assignment: *cryptographic algorithm*]
[33] [assignment: *cryptographic key sizes*]
[34] [assignment: *list of standards*]
[35] [assignment: *a defined quality metric*]

source, the internal state of the RNG shall have at least 148 bit of entropy.

(DRG.3.2) The RNG provides forward secrecy.

(DRG.3.3) The RNG provides backward secrecy even if the current internal state is known.

(DRG.3.4) The RNG, initialized with a random seed after start-, generates output for which in 3DES mode 2^{35} strings of bit length 128 are mutually different with probability at least $1 – 2^{-17}$ in 3DES mode.

(DRG.3.5) Statistical test suites cannot practically distinguish the random numbers from output sequences of an ideal RNG. The random numbers must pass test procedure A.

6.1.2 Identification and Authentication

FIA_AFL.1/PIN Authentication failure handling – eHC-PIN

Hierarchical to:	No other components.
FIA_AFL.1.1/PIN	The TSF shall detect when *3*[36] unsuccessful authentication attempts occur related to consecutive failed human user authentication for the health care application[37].
FIA_AFL.1.2/PIN	When the defined number of unsuccessful authentication attempts has been *met*[38], the TSF shall block the PIN for authentication until successful unblock with resetting code[39].
Dependencies:	FIA_UAU.1 Timing of authentication.

FIA_AFL.1/PUC Authentication Failure Handling – eHC-PIN-unblocking code

Hierarchical to:	No other components.
FIA_AFL.1.1/PUC	The TSF shall detect when *10*[40] ~~unsuccessful~~[41] attempts occur related to usage of the eHC-PIN unblocking code[42].

[36] [selection: [assignment: *positive integer number*], *an administrator configurable positive integer within* [assignment*: range of acceptable values*]]

[37] [*assignment: list of authentication events*]

[38] [selection: *met or surpassed*]

[39] [assignment: *list of actions*]

[40] *[selection: [assignment: positive integer number], an administrator configurable positive integer within [assignment: range of acceptable values]]*

[41] refinement: not only unsuccessful but all attempts shall be counted here – obviously this refinement is valid, because the original requirement is still fulfilled

[42] [assignment: *list of authentication events*]

FIA_AFL.1.2/PUC When the defined number of ~~unsuccessful~~[43] authentication attempts has been *met*[44], the TSF shall
- warn the entity connected
- not unblock the referenced blocked PIN
- block the PUC resp. the verification mechanism for this PUC such that any subsequent authentication attempt with this PUC will fail and an unblocking of all blocked PINs related to this PUC is no longer possible
- be able to indicate to subsequent users the reason for the blocking of the PUC[45].

Dependencies: FIA_UAU.1 Timing of authentication

FIA_ATD.1 User attributes definition

Hierarchical to: No other components.

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users: identity and role[46].

Dependencies: No dependencies.

FIA_UID.1 Timing of identification

Hierarchical to: No other components.

FIA_UID.1.1 The TSF shall allow

(1) reading the ATR,
(2) reading the Card Verifiable Authentication Certificate,
(3) reading the Certificate Service Provider Certificate,
(4) *execution of commands allowed without preceding successful authentication due to the access rules set*[47]

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.

[43] refinement: not only unsuccessful but all shall be counted here – obviously this refinement is valid, because the original requirement is still fulfilled

[44] [selection: *met or surpassed*]

[45] [assignment: *list of actions*] with refinement of the list of actions *which at least includes: block the PIN unblocking code* – obviously this refinement is valid, because the original requirement is still fulfilled

[46] [assignment: *list of security attributes*]

[47] [assignment: *list of TSF-mediated actions*]

Dependencies: No dependencies.

FIA_UAU.1 Timing of authentication

Hierarchical to: No other components.

FIA_UAU.1.1 The TSF shall allow

(1) reading the ATR
(2) reading the Card Verifiable Authentication Certificate
(3) reading the Certificate Service Provider self-signed Certificate
(4) Identification by providing the users eHC-PIN
(5) identification by providing the users certificate
(6) *execution of commands allowed without preceding successful authentication due to the access rules set*][48]

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.

Dependencies: FIA_UID.1 Timing of identification

FIA_UAU.4 Single-use authentication mechanisms

Hierarchical to: No other components.

FIA_UAU.4.1 The TSF shall prevent reuse of authentication data related to Card-to-Card Authentication Mechanism[49].

Dependencies: No dependencies.

6.1.3 Access Control

The Security Function Policy (SFP) **SFP_access_rules,** which as defined in the security objective OT.Access_Rights (section 4.1.1), is used in the requirements “Complete Access Control (FDP_ACC.2)”, “Security attribute based access control (FDP_ACF.1)”, “Basic data exchange confidentiality (FDP_UCT.1)” and “Basic data exchange confidentiality (FDP_UCT.1)”.

The access control policy **SFP_access_rules** is only defined for the End Usage phase of the TOE. Note, that access rules for initialisation and personalisation phases are defined by management SFRs (FMT_MTD.1, see section 6.1.5), not by an explicit policy.

The following SFRs require the TOE to enforce the security policy **SFP_access_rules**. Note that all subjects, objects, security attributes, access methods and access rules are defined

[48] [assignment: *list of TSF-mediated actions*]

[49] [assignment: *identified authentication mechanism(*s)]

already in this policy, which is described in detail in the PP. Therefore all of the following SFRs simply refer to this policy in all assignments.

FDP_ACC.2 Complete Access Control

Hierarchical to: FDP_ACC.1 Subset access control

FDP_ACC.2.1 The TSF shall enforce the SFP_access_rules[50] on all subjects and objects defined by SFP_access_rules[51] 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.

Dependencies: FDP_ACF.1 Security attribute based access control

FDP_ACF.1 Security attributes based access control

Hierarchical to: No other components.

FDP_ACF.1.1 The TSF shall enforce the SFP_access_rules[52] to objects based on the following: all subjects and objects together with their respective security attributes as defined in SFP_access_rules[53].

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: rules for all access methods and the access rules defined in SFP_access_rules[54].

FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: none[55].

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the following additional rules: rules for all access methods and the access rules defined in SFP_access_rules[56].

Dependencies: FDP_ACC.1 Subset access control
FMT_MSA.3 Static attribute initialization

FDP_RIP.1 Residual Information Protection

[50] [assignment*: access control SFP*]

[51] [assignment: *list of subjects and objects*]

[52] [assignment: *access control SFP*]

[53] [assignment: *list of subjects and objects controlled under the indicated SFP, and. for each, the SFP-relevant security attributes*, *or named groups of SFP-relevant security attributes*]

[54] [assignment: *rules governing access among controlled subjects and controlled objects using controlled operations on controlled objects*]

[55] [assignment: *rules, based on security attributes, that explicitly authorise access of subjects to objects*]

[56] [assignment: *rules, based on security attributes, that explicitly deny access of subjects to objects*]

Hierarchical to: No other components.

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*[57] the following objects: *security relevant material (as secret and private cryptographic keys, PINs, PUCs, data in all files which are not freely accessible ...)*[58].

Dependencies: No dependencies.

FDP_SDI.2 Stored Data Integrity

Hierarchical to: FDP_SDI.1 Stored data integrity monitoring

FDP_SDI.2.1 The TSF shall monitor user data stored in containers controlled by the TSF for integrity errors[59] on all objects, based on the following attributes: *checksum secured persistently stored on the following data:*

- *PINs,*
- *cryptographic keys,*
- *security relevant status variables of the card (e.g. authentication status for the PIN or for mutual authenticate),*
- *input data for electronic signatures,*
- *user data in files on the card,*
- *file management information (like access rules for files), and*
- *the card life cycle status*[60].

FDP_SDI.2.2 Upon detection of a data integrity error, the TSF shall

1. prohibit the use of the altered data
2. inform the connected entity about integrity error[61].

Dependencies: No dependencies.

[57] [selection: *allocation of the resource to, deallocation of the resource from]*

[58] [*assignment: list of objects*] with refinement of the list of objects *at least including: PINs, secret and private cryptographic keys, data in all files, which are not freely accessible* – obviously this refinement is valid, because the original requirement is still fulfilled

[59] *[assignment: integrity errors]*

[60] *[assignment: user data attributes]* with refinement of the list of user data attributes *the attributes shall be chosen in a way that at least the following data are included: PINs, cryptographic keys, security relevant status variables of the card (e.g. authentication status for the PIN or for mutual authenticate), input data for electronic signatures, user data in files on the card, file management information (like access rules for files), and the card life cycle status*– obviously this refinement is valid, because the original requirement is still fulfilled

[61] *[assignment: action to be taken]*

6.1.4 Inter-TSF-Transfer

Application note 35: FDP_UCT.1, FDP_UIT.1 and FTP_ITC.1 require the TOE to protect User Data transmitted between the TOE and a connected device by secure messaging with encryption and message authentication codes after successful authentication of the remote device. The authentication mechanisms as part of the Card-to-Card Authentication Mechanism include the key agreement for the encryption and the message authentication key to be used for secure messaging. The rules for the data transfer are defined in the security policy **SFP_access_rules** defined in objective OT.Access_Rights (section 4.1.1).

FDP_UCT.1 Basic data exchange confidentiality

Hierarchical to: No other components.

FDP_UCT.1.1 The TSF shall enforce the SFP_access_rules[62] to transmit and receive[63] user data in a manner protected from unauthorised disclosure.

Dependencies: [FTP_ITC.1 Inter-TSF trusted channel, or
FTP_TRP.1 Trusted path]
[FDP_ACC.1 Subset access control, or
FDP_IFC.1 Subset information flow control]

FDP_UIT.1 Data exchange integrity

Hierarchical to: No other components.

FDP_UIT.1.1 The TSF shall enforce the SFP_access_rules[64] to transmit and receive[65] user data in a manner protected from modification, deletion, insertion and replay[66] errors.

FDP_UIT.1.2 The TSF shall be able to determine on receipt of user data, whether modification, deletion, insertion and replay[67] has occurred.

Dependencies: [FDP_ACC.1 Subset access control, or
FDP_IFC.1 Subset information flow control]
[FTP_ITC.1 Inter-TSF trusted channel, or
FTP_TRP.1 Trusted path]

FTP_ITC.1 Inter-TSF Trusted Channel

Hierarchical to: No other components.

[62] [assignment: *access control SFP(s) and/or information flow control SFP(s)*]

[63] [selection*: transmit, receive*]

[64] [assignment: *access control SFP(s) and/or information flow control SFP(s)*]

[65] [selection*: transmit, receive*]

[66] [selection: *modification, deletion, insertion, replay*]

[67] [selection: *modification*, *deletion, insertion, replay*]

FTP_ITC.1.1 The TSF shall provide a communication channel between itself and another trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure.

FTP_ITC.1.2 The TSF shall permit another trusted IT product[68] to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for all functions requiring a trusted channel as defined by SFP_access_rules[69].

Dependencies: No dependencies.

6.1.5 Security Management

FMT_SMF.1 Specification of Management Functions

Hierarchical to: No other components.

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions:

1. Initialization
2. Personalization
3. the "Service_Card_Management"
4. Modification of the PIN[70].

Dependencies: No Dependencies

FMT_SMR.1 Security roles

Hierarchical to: No other components.

FMT_SMR.1.1 The TSF shall maintain the roles Health Professional, Medical Assistant, Security Module Card (health care), Self Service Terminal, Health Insurance Agency Service Provider, Combined Services Provider, Cardholder, Download Service Provider, Personalisation Service Provider, TOE Manufacturer[71].

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

Dependencies: FIA_UID.1 Timing of identification

FMT_LIM.1 Limited capabilities

[68] [selection: *the TSF, the another trusted IT product*]

[69] [assignment: *list of functions for which a trusted channel is required*].

[70] [assignment: *list of security management functions to be provided by the TSF*]

[71] [assignment: *the authorised identified roles*]

Hierarchical to: No other components.

FMT_LIM.1.1 The TSF shall be designed in a manner that limits their capabilities so that in conjunction with "Limited availability (FMT_LIM.2)" the following policy is enforced: Deploying Test Features after TOE Delivery does not allow User Data to be disclosed or manipulated, TSF data to be disclosed or manipulated, software to be reconstructed and no substantial information about construction of TSF to be gathered which may enable other attacks[72].

Dependencies: FMT_LIM.2 Limited availability.

FMT_LIM.2 Limited availability

Hierarchical to: No other components.

FMT_LIM.2.1 The TSF shall be designed in a manner that limits their availability so that in conjunction with "Limited capabilities (FMT_LIM.1)" the following policy is enforced: Deploying Test Features after TOE Delivery does not allow User Data to be disclosed or manipulated, TSF data to be disclosed or manipulated, software to be reconstructed and no substantial information about construction of TSF to be gathered which may enable other attacks[73].

Dependencies: FMT_LIM.1 Limited capabilities.

FMT_MTD.1/Ini Management of TSF data - Initialisation

Hierarchical to: No other components.

FMT_MTD.1.1/Ini The TSF shall restrict the ability to write[74] the initialisation data[75] to the TOE Manufacturer[76].

Dependencies: FMT_SMF.1 Specification of management functions
FMT_SMR.1 Security roles

FMT_MTD.1/Pers Management of TSF data - Personalisation

Hierarchical to: No other components.

FMT_MTD.1.1/Pers The TSF shall restrict the ability to write[77] the personalisation data[78] to the Personalisation Service Provider[79].

[72] [assignment: *Limited capability and availability policy*]

[73] [assignment: *Limited capability and availability policy*]

[74] [selection: *change default, query, modify, delete, clear*, [assignment: *other operations*]]

[75] [assignment: *list of TSF data*]

[76] [assignment: *the authorised identified roles*]

[77] [selection: *change default, query, modify, delete, clear*, [assignment: *other operations*]]

Dependencies: FMT_SMF.1 Specification of management functions
FMT_SMR.1 Security roles

FMT_MTD.1/CMS Management of TSF data – Card Management

Hierarchical to: No other components.

FMT_MTD.1.1/CMS The TSF shall restrict the ability to write[80] the

1. File structures for additional Applications,
2. Cryptographic Keys for additional applications,
3. PINs and other user authentication reference data for additional applications and
4. Access Rights for additional applications[81]

to the Download Service Provider[82].

Dependencies: FMT_SMF.1 Specification of management functions
FMT_SMR.1 Security roles

FMT_MTD.1/PIN Management of TSF data – Human User Authentication data

Hierarchical to: No other components.

FMT_MTD.1.1/PIN The TSF shall restrict the ability to modify and unblock[83] the PIN[84] to the Cardholder[85].

Dependencies: FMT_SMF.1 Specification of management functions
FMT_SMR.1 Security roles

FMT_MTD.1/KEY_MOD Management of TSF data – Key Management

Hierarchical to: No other components.

FMT_MTD.1.1/KEY_MOD The TSF shall restrict the ability to modify[86] the Public Key for CV Certification Verification[87] to none[88].

---

[78] [assignment: *list of TSF data*]

[79] [assignment: *the authorised identified roles*]

[80] [selection: *change default, query, modify, delete, clear*, [assignment: *other operations*]]

[81] [assignment: *list of TSF data*]

[82] [assignment: *the authorised identified roles*]

[83] [selection: *change default, query, modify, delete, clear*, [assignment: *other operations*]]

[84] [assignment: *list of TSF data*]

[85] [assignment: *the authorised identified roles*]

[86] [selection: *change default, query, modify, delete, clear*, [assignment: *other operations*]]

[87] [assignment: *list of TSF data*]

[88] [assignment: *the authorised identified roles*]

Dependencies: FMT_SMF.1 Specification of management functions
FMT_SMR.1 Security roles

6.1.6 General Security Functions

FPT_EMSEC.1 TOE Emanation

Hierarchical to: No other components.

FPT_EMSEC.1.1 The TOE shall not emit *information on IC power consumption, information on command execution time, information on electromagnetic emanations*[89] in excess of *non-useful information*[90] enabling access to

1. PIN and PUC[91]

and

2. Card Authentication Private Keys,
3. Client-Server Authentication Private Key,
4. Document Cipher Key Decipher Key,
5. secure messaging keys[92].

FPT_EMSEC.1.2 The TSF shall ensure any user[93] are unable to use the following interface smart card circuit contacts[94] to gain access to

1. PIN and PUC[95]

and

2. Card Authentication Private Key,
3. Client-Server Authentication Private Key
4. Document Cipher Key Decipher Key
5. secure messaging keys[96].

Dependencies: No other components.

FPT_FLS.1 Failure with preservation of secure state

[89] [*assignment: types of emissions*]
[90] [assignment: *specified limits*]
[91] [assignment: *list of types of TSF data*]
[92] [assignment: *list of types of user data*]
[93] [assignment: *type of users*]
[94] [assignment: *type of connection*]
[95] [assignment: *list of types of TSF data*]
[96] [assignment: *list of types of user data*]

Hierarchical to: No other components.

FPT_FLS.1.1 The TSF shall preserve a secure state when the following types of failures occur:

1. exposure to operating conditions where therefore a malfunction could occur,
2. self-test according to FPT_TST.1[97].

Dependencies: No dependencies

FPT_PHP.3 Resistance to physical attack

Hierarchical to: No other components.

FPT_PHP.3.1 The TSF shall resist physical manipulation and physical probing[98] to the TSF[99] by responding automatically such that the SFRs are always enforced.

Dependencies: No dependencies.

FPT_TST.1 TSF testing

Hierarchical to: No other components.

FPT_TST.1.1 The TSF shall run a suite of self tests *during initial start-up, periodically during normal operation*[100] to demonstrate the correct operation of *the TSF*[101].

FPT_TST.1.2 The TSF shall provide authorised users with the capability to verify the integrity of *TSF data*[102].

FPT_TST.1.3 The TSF shall provide authorised users with the capability to verify the integrity of stored *TSF executable code*[103].

Dependencies: No dependencies

[97] [assignment: *list of types of failures in the TSF*]

[98] [assignment: *physical tampering scenarios*]

[99] [assignment: *list of TSF devices/elements*]

[100] [selection: *during initial start-up, periodically during normal operation, at the request of the authorised user, at the conditions* [*assignment: conditions under which self test should occur*]]

[101] [selection: *[assignment: parts of TSF], the TSF]*

[102] [selection: *[assignment: parts of TSF data], TSF data]*

[103] [selection: *[assignment: parts of TSF]*, *TSF*]

6.2 Security Assurance Requirements for the TOE

1 The assurance components for the evaluation of the TOE and its development and operating environment are those taken from the

Evaluation Assurance Level 4 (EAL4)

and augmented by taking the following components:

AVA_VAN.5.

6.3 Security Requirements Rationale

The following chapter covers the security objectives rationale, the security requirements rationale.

The chapter is not disclosed in the ST-Lite.

7 TOE Summary Specification

7.1 TOE Security Functions

7.1.1 TOE Security Functions / TOE_IC

For the definition of the TOE Security Functions (TSF) related to the TOE-IC refer to the Security Targets /ST_IC/, Chapter 7.1 and /ST-IC+CL/, Chapter 7.

The TSFs defined for the TOE-IC cover the following functions which are relevant for the TOE: SS.RNG, SS.HW_DES, SF.OPC, SF.PHY, SF.LOG, SF.COMP, SF.MEM_ACC, SF.SFR_ACC, SF.FFW, SF.FIRMWARE, SS.RSA, SS.SHA, SS.SW.RNG, SS.Object_Reuse, SS.COPY, SS.DES, SS.RSA_PublicExp and SS.RSA_KeyGen.

7.1.2 TOE Security Functions / TOE_ES

The following section gives a survey of the TSFs of the TOE´s Smartcard Embedded Software.

TOE Security Functions / TOE_ES	
Access Control	
F.ACS_SFP	**Security Attribute Based Access Control**
	The TSF enforces the SFPs **SFP_access_rules** as defined in Chapter 4.1.1 in the PP. The TSF controls the access to data stored in the TOE and to functionality provided by the TOE. The access control is realised by usage of access rules as security attributes. Access to a DF, an EF, a key, a PIN or other user data is only possible if the related access rule is fulfilled. In particular, the TSF checks prior to command execution if the command specific requirements concerning user authentication and secure communication are satisfied.
Identification and Authentication	
F.IA_AKEY	**Key Based User / TOE Authentication Based on Asymmetric Cryptography**
	The TSF provides the functionality of a key based external and internal authentication on the base of asymmetric cryptography. By an external authentication, users of the TOE can be authenticated with regard to the TOE. Vice versa, by an internal authentication, the TOE itself can be authenticated with regard to the external world. Both authentication mechanisms base on a challenge-response procedure using random numbers. The TSF enforces the following different internal and external authentication mechanisms: - Internal authentication without session key agreement according to /ISO 9796-2/, /EXS_EHC_1/, Chapters 7 and 15 - External authentication without session key agreement according to /ISO 9796-2/, /EXS_EHC_1/, Chapters 7 and 15 - Internal authentication including one step of session key and send sequence counter agreement according to /ISO 9796-2/, /EXS_EHC_1/, Chapters 7 and

	15 - External authentication including one step of session key and send sequence counter agreement according to /ISO 9796-2/, Chapters 7 and 15 - Internal authentication according to /EXS_EHC_1/, Chapters 7 and 15 Note: Each external authentication process requires a preceding Get Challenge – operation. The private and public keys necessary on the card´s side for authentication purposes are either generated on-card (with support by the TSF SS.RSA_KeyGen) or imported during the initialisation, personalisation or end-usage phase of the TOE. In particular, the import of public keys can be performed in the form of CV certificates what is connected with the verification of the respective CV certificate under usage of the TSF F.VER_DIGSIG. In each case, the keys involved on the card´s side in the authentication processes have to be explicitly referenced prior to their usage. The access to the keys necessary for the authentication processes is controlled by the specific SFP which is defined for the respective application using the authentication keys. The execution of the specific SFP is task of the TSF F.ACS_SFP for access control. In the case of a successful external authentication attempt the TSF sets a corresponding actual security state for key based user authentication. The TSF makes use of asymmetric cryptography with generation and verification of RSA digital signatures resp. RSA encryption and decryption and is therefore directly connected with the TSF F.CRYPTO. Depending on the type of authentication mechanism, the combination of a successful internal and external authentication process can include the generation of session keys (incl. send sequence counter). Depending on the type of authentication mechanism, the TSF stores the generated session keys volatile and on demand as well persistently on the card. The generated keys can be used for securing the following data exchange between the TOE and the external world (in the current or a later session) with the objective of data confidentiality and data integrity and authenticity (Secure Messaging). In addition, as well depending on the type of authentication mechanism, the generated keys can be used further on for authentication processes based on symmetric cryptography.
F.IA_SKEY	**Key Based User / TOE Authentication Based on Symmetric Cryptography**
	The TSF provides the functionality of a key based external and internal authentication on the base of symmetric cryptography. By an external authentication, users of the TOE can be authenticated with regard to the TOE. Vice versa, by an internal authentication, the TOE itself can be authenticated with regard to the external world. Both authentication mechanisms base on a challenge-response procedure using random numbers. The TSF enforces the following different internal and external authentication mechanisms: - Internal authentication with / without individual key derivation and without session key generation according to /EXS_EHC_1/, Chapters 7 and 15, /ISO 9796-2/ - External authentication with / without individual key derivation and without session key generation according to /EXS_EHC_1/, Chapters 7 and 15, /ISO 9796-2/ - Mutual authentication with / without individual key derivation and without session key generation according /EXS_EHC_1/, Chapters 7 and 15, /ISO 9796-2/ - Internal authentication with / without individual key derivation and including the first step of session key and send sequence counter generation according to /EXS_EHC_1/, Chapters 7 and 15, /ANSI X9.63/, /ISO 9796-2/

	- External authentication with / without individual key derivation and including the last step of session key and send sequence counter generation according to /EXS_EHC_1/, Chapters 7 and 15, /ANSI X9.63/, /ISO 9796-2/ - Mutual authentication with / without individual key derivation and including session key and send sequence counter generation according to /EXS_EHC_1/, Chapters 7 and 15, /ANSI X9.63/, /ISO 9796-2/ Note: Each external authentication process requires a preceding Get Challenge – operation. The symmetric keys necessary on the card´s side for the authentication mechanisms can either be generated on-card by a derivation process for deriving individual keys before the main authentication process starts. This key derivation process is performed by the TSF F.CRYPTO. Alternatively, symmetric keys imported during the initialisation, personalisation or end-usage phase of the TOE or agreed within a preceding authentication process can be used. The access to the keys necessary for the authentication processes is controlled by the specific SFP which is defined for the respective application using the authentication keys. The execution of the specific SFP is task of the TSF F.ACS_SFP for access control. In the case of a successful external authentication attempt the TSF sets a corresponding actual security state for key based user authentication. The TSF makes use of symmetric cryptography with DES based encryption, decryption, MAC generation resp. MAC verification. Hence, the TSF F.IA_SKEY is directly connected with the TSF F.CRYPTO. Depending on the type of authentication mechanism, the combination of a successful internal and external authentication process can include the generation of session keys (incl. send sequence counter). Depending on the type of authentication mechanism, the TSF stores the generated session keys volatile and on demand as well persistently on the card. The generated keys can be used for securing the following data exchange between the TOE and the external world (in the current or a later session) with the objective of data confidentiality and data integrity and authenticity (Secure Messaging). In addition, as well depending on the type of authentication mechanism, the generated keys can be used further on for authentication processes based on symmetric cryptography.
F.IA_PWD	**Password Based User Authentication**
	Users of the TOE can be authenticated (towards the TOE) by means of a card holder authentication process. For the card holder authentication process, the TSF compares the card holder verification information, here a password (PIN), provided by a subject with a corresponding secret reference value stored permanently on the card. The TSF uses for the authentication process the password referenced by the external world. The access to the relevant password resp. its reference value is controlled by the specific SFP which is defined for the respective application using the password. The execution of the specific SFP is task of the TSF F.ACS_SFP for access control. The card holder authentication process can be performed by usage of the command Verify or Change Reference Data (whereat the latter command makes a password change possible). Each password used for authentication purposes is connected with an own error usage counter and an own usage counter. Furthermore, each password is connected with an own resetting code (PUC) whereat the resetting code itself is connected with an own usage counter (but no error usage counter). The number of applications of a password for authentication purposes with the command Verify is limited by its usage counter. The TSF allows at maximum for a number of authentication attempts with a password as restricted by its usage counter. The value for the usage counter can be predefined as infinite, i.e. the password can be used without any

	limit. A password with an expired usage counter cannot be longer used for authentication purposes with the command Verify (but with the command Change Reference Data). In the case of a password with a finite usage counter, each authentication attempt with the command Verify decrements the usage counter of the password, independently whether the authentication attempt succeeds or fails. A successful authentication attempt with the command Change Reference Data re-initialises the usage counter to its predefined initial value. The TSF detects for a password when a predefined number of consecutive unsuccessful authentication attempts occurs related to the card holder authentication process. Each consecutive unsuccessful comparison of the presented password with the reference value stored on the card is recorded by the TSF in order to limit the number of further authentication attempts with this password. In the case of a successful authentication attempt a corresponding actual security state for the password is set and the error usage counter of the password is re-initialised to its predefined initial value. If an authentication attempt with the password fails, the corresponding actual security state is reset and the error usage counter of the password is decreased. When the defined maximum number of unsuccessful authentication attempts has been met or surpassed, the TSF blocks the corresponding password for any further authentication attempt. A password with an expired error usage counter can be unblocked by usage of the related resetting code, provided that the usage counters of the password and of the resetting code are not expired. Otherwise, there is no way to unblock the password so that this password is invalid for each further authentication attempt. The unblocking of a blocked password can be performed by usage of the command Reset Retry Counter only. In the case of a successful authentication attempt with the resetting code related to the blocked password, the expired error usage counter is re-initialised to its initial value (as well as for the usage counter of the password) and hence, the password can be used further on for authentication attempts. The number of applications of a resetting code for authentication purposes is limited by its usage counter. The TSF allows at maximum for a number of authentication attempts with the resetting code as restricted by its usage counter. Each unblocking attempt with the command Reset Retry Counter decrements the usage counter of the resetting code, independently whether the authentication attempt with the resetting code succeeds or fails. The unblocking process for a blocked password can be combined with a change of this password. However, even if the command Reset Retry Counter resp. the authentication with the resetting code succeeds, the actual security state for the password will not be set. For security reasons, a password shall be connected with an error usage counter with a sufficiently small value as initial value. Furthermore, the usage of the related resetting code itself shall be limited by a usage counter with a sufficiently small initial value. In general, a security state set due to a successful authentication attempt can be valid for several following TOE commands. However, as well, it is possible to restrict the validity of such an authentication state to one single following TOE command, i.e. after the next command has accessed this security state it will be reset by the TSF. The TSF does not check the quality of passwords or resetting codes used. The sufficient quality of passwords and resetting codes lies in the responsibility of the external world only. The transfer of passwords and resetting codes to the TOE can be executed in unsecured mode, i.e. without usage of Secure Messaging, or alternatively in secured mode, i.e. with usage of Secure Messaging. In the latter case, the TSFs F.EX_CONF and F.EX_INT are involved. For the TOE´s eHC Application, the concrete usage of PIN and PUK, in particular the

	definition of error usage counters and usage counters and their initial values, the (minimal) lengths of PIN and PUK and the access to the commands Verify, Change Reference Data and Reset Retry Counter is regulated by the specification/EXS_EHC_2/.
Integrity of Stored Data	
F.DATA_INT	**Stored Data Integrity Monitoring and Action**
	The TSF monitors data stored within the TOE for integrity errors. This concerns all - DFs - EFs - Passwords incl. related attributes - Cryptographic keys incl. related attributes - Security critical data stored within the card and channel context (session keys incl. attributes, status information as actual security states for key and password based authentication, hash values, further security relevant card and channel information) The monitoring is based on the following attributes: - Checksum (CRC) attached to the header of a file - Checksum (CRC) attached to the data body of a file - Checksums (CRC) attached to each secret (password, cryptographic key) and its related attributes stored in the EEPROM - Checksums (CRC) attached to card and channel context related security critical information Each access of the TOE to a DF, to an EF, to a secret (password or cryptographic key incl. its related attributes) or to security critical card resp. channel context data the TSF is secured with an integrity check on base of the mentioned attributes. Upon detection of a data integrity error, the TSF informs the user about this fault (output of a warning). If the checksum of the header of a file has been detected as corrupted, the data contained in the affected file are no longer accessible. If the data contained in a file are not of integrity, the affected data will be treated in the following way: - For the Read access, the affected data will be exported, but the data export will be connected with a warning. - For the Update access, the integrity error of the affected data will be ignored, and the data imported by the command will be stored and a new checksum will be computed. - For all remaining access modes, the affected data will not be used for data processing. If a secret (password, cryptographic key) and its related attributes are corrupted, the secret and its related data will not be processed. If security critical card or channel context data are not of integrity, the Smartcard Embedded Software immediately jumps into an endless-loop (re-activation by reset possible).
Data Exchange	
F.EX_CONF	**Confidentiality of Data Exchange**
	The TSF provides the capability to ensure that secret data which is exchanged between the TOE and the external world remains confidential during transmission. For this purpose, encryption based on symmetric cryptography is applied to the secret data.

	The TSF ensures that the user and the user data's access condition have indicated confidentiality for the data exchange. Securing the data transfer with regard to data confidentiality is done by Secure Messaging according to the standard ISO/IEC 7816-4. The cryptographic key used for securing the data transfer is either a symmetric session or static key. In case of a session key, the key is negotiated during a preceding mutual authentication process (based on a random challenge and response procedure) between the TOE and the external world (realised by the TSFs F.IA_SKEY, F.IA_AKEY, and F.CRYPTO). For encryption and decryption, the TSF makes use of the TSF F.CRYPTO for DES functionality.
F.EX_INT	**Integrity and Authenticity of Data Exchange**
	The TSF provides the capability to ensure that data which is exchanged between the TOE and the external world remains integer and authentic during transmission. For this purpose, cryptographic checksums based on symmetric cryptography are applied to the data. The TSF ensures that the user and the user data's access condition have indicated integrity and authenticity for the data exchange. Securing the data transfer with regard to data integrity and authenticity is done by Secure Messaging according to the standard /ISO_7816_4/. The cryptographic key used for securing the data transfer is either a symmetric session or static key. In case of a session key, the key is negotiated during a preceding mutual authentication process (based on a random challenge and response procedure) between the TOE and the external world (realised by the TSFs F.IA_SKEY, F.IA_AKEY, and F.CRYPTO). For checksum securing and verification, the TSF makes use of the TSF F.CRYPTO for DES functionality.
Object Reuse	
F.RIP	**Residual Information Protection**
	The TSF ensures that any previous information content of a resource is explicitly erased upon the deallocation of the resource used for any of the following components: - All volatile and non-volatile memory areas used for operations in which security relevant material (as e.g. cryptographic data, passwords or other security critical data) is involved. Explicit erasure is defined as physical erasure. The TSF is supported by the TSF SF.Object_Reuse of the underlying IC and its Dedicated Support Software.
Protection	
F.FAIL_PROT	**Hardware and Software Failure Protection**
	The TSF preserves a secure operation state of the TOE when the following types of failures and attacks occur: - HW and/or SW induced reset - Power supply cut-off - Power supply variations - Unexpected abortion of the execution of the TSF due to external or internal events (in particular, break of a transaction before completion)

	- System breakdown - Internal HW and/or SW failure - Manipulation of executable code - Corruption of status information (as e.g. card status information, object life cycle state, actual security state related to key and password based authentication, ...) - Environmental stress - Input of inconsistent or improper data - Tampering - Manipulation resp. insufficient quality of the HW-RNG The TSF makes use of HW and SW based security features and corresponding mechanisms to monitor and detect induced HW and SW failures and tampering attacks. In particular, the TSF is supported by the IC specific TSFs SF.OPC and SF.PHY. Upon the detection of a failure of the above mentioned type the TSF reacts in such a way that the TSP is not violated. The TOE changes immediately to a locked state and cannot be used any longer within the actual session. Depending on the type of the detected attack to the underlying IC (incl. its Dedicated Software) or to the Smartcard Embedded Software code the TOE will be irreversible locked resp. can be reactivated by a reset.
F.SIDE_CHAN	**Side Channel Analysis Control**
	The TSF provides suitable HW and SW based mechanisms to prevent attacks by side channel analysis like Simple Power Analysis (SPA), Differential Power Analysis (DPA), Differential Fault Analysis (DFA) and Timing analysis (TA). The TSF ensures that all countermeasures available are used in such a way that they support each other. In particular, the TSF is supported by the TSF SF.LOG of the underlying IC and its Dedicated Support Software. The TSF acts in such a manner that all security critical operations of the TOE, in particular the TOE´s cryptographic operations, are suitably secured by these HW and SW countermeasures. The TSF guarantees that information on IC power consumption, information on command execution time and information on electromagnetic emanations do not lead to useful information on processed security critical data as secret cryptographic keys or passwords. In particular, the IC contacts as Vcc, I/O and GND or the IC surface do not make it possible for an attacker to gain access to security critical data as secret cryptographic keys or passwords. The TSF enforces the installation of a secure session before any cryptographic operation is started. In particular, the installation of a secure session does not only concern the core cryptographic operation itself. All preparing security relevant actions performed prior to the core cryptographic operation as e.g. the generation of session keys, the process of loading keys into the dedicated IC cryptographic modules and the data preparation as reformatting or padding are involved as well. Furthermore, the secure session covers all security relevant actions which follow the core cryptographic operation as e.g. the post-processing of the output data.
F.SELFTEST	**Self-Test**
	The TSF covers different types of self tests whereat each self-test consists of a check of a dedicated integrity attribute related to (parts of) the TOE´s code resp. data. The TSF integrates self-tests with the following objectives: The TSF provides the capability of conducting a self-test during initial start-up, i.e. after each reset, to demonstrate the correct operation of its TSFs. This self-test is performed automatically by the TOE and consists of the verification of the integrity of any software

	code stored in the EEPROM area. Furthermore, the TSF provides authorised users - here the Smartcard Embedded Software of the TOE (TOE_ES) itself - with the capability to verify the integrity of TSF data during run-time. The self-test is performed automatically by the TOE and is supported by the TSF F.DATA_INT. Additionally, the TSF provides authorised users with the capability to verify the integrity of stored TSF executable code. This concerns only the production phase, more precise the initialisation phase of the TOE (phase 5 of the product´s life cycle). Prior to the initialisation of the TOE, the ROM-code of the TOE can be verified on demand by the Smartcard Embedded Software developer. The integrity of the whole EEPROM-code is checked automatically by the TOE during the storage of the initialisation file in the framework of the TOE´s initialisation. These self-tests are supported by the TSF F.CRYPTO (SHA-1 hash value calculation, MAC verification). The TSF supports all other TSFs defined for the Smartcard Embedded Software (TOE_ES).
Cryptographic Operations	
F.CRYPTO	**Cryptographic Support**
	The TSF provides cryptographic support for the other TSFs using cryptographic mechanisms. The TSF supports: - DES/3DES algorithm according to the standard /FIPS 46-3/ resp. /ANSI X9.52/ with a key length of 168 bit (used for encryption, decryption, MAC generation and verification according to /FIPS 46-3/, /ANSI X9.52/, /ANSI X9.19/ , Chapter 7) - RSA core algorithm according to the standard /PKCS1/ with key lengths of 2048 bit modulus lengths (used for RSA encryption, decryption, signature generation and verification, see other TSFs related to RSA based mechanisms) - Random number generation by a pseudo RNG. The generator is seeded by the hardware random number generator (see /CL_UG/, /CL_UG_RND/) - SHA-256 hash value calculation according to /ALGCAT/, Chapter 2 - Negotiation of 3DES session keys The resistance of the TSF against SPA, DPA, DFA and TA is part of the TSF F.SIDE_CHAN. The random number generation is in particular used for RSA and DES key generation and authentication mechanisms. The mechanism for the generation of session keys is directly connected with the TSFs F.IA_AKEY and F.IA_SKEY which realise internal and external authentication processes. Furthermore, the generation of random numbers of high quality, and depending on the authentication type, the SHA-256 hash value calculation of TSF F.CRYPTO are involved. The TSF is directly supported by the TSFs of the underlying IC and its Cryptographic Library which supply cryptographic functionality. In particular, the TSFs SS.RNG, SS.HW_DES, SS.DES, SS.RSA, SS.RSA_KeyGen, SS.RSA_PublicExp and SS.SW_RNG are involved.
F.GEN_DIGSIG	**RSA Generation of Digital Signatures**
	The TSF provides digital signature functionality based on asymmetric cryptography, particularly based on the RSA algorithm with key lengths of 2048 bit modulus length. The TSF digital signature function will be used for several purposes with different formats

	for the digital signature input: - Explicit generation of digital signatures using the signature scheme with appendix according to the standard /PKCS1/, Chapter 8.2.1 and with hash algorithm SHA-2 (256 bit), see /EXS_EHC_1/, Chapter 7 - Explicit generation of digital signatures using the signature scheme with appendix according to the standard /ISO 9796-2/ with random number based on the hash algorithm SHA-2 (224, 256, 384 resp. 512 bit) resp. RIPEMD160 (external hash value calculation), see /EXS_EHC_1/, Chapter 7 - Implicit generation of digital signatures within authentication mechanisms for the creation of authentication tokens using the signature scheme with message recovery according to the standard /ISO 9796-2/ based on the hash algorithm SHA-256, see /EXS_EHC_1/, Chapters 7 and 16 - Implicit generation of digital signatures within authentication mechanisms for the creation of authentication tokens using the signature scheme with message recovery according to the standard /PKCS1/, Chapter 8.2.1 without hash and OID, but with an additional limitation of the length of the input message, see /EXS_EHC_1/, Chapters 7 and 16 The TSF function for generation of a digital signature uses the private key which has been referenced before. The random numbers necessary for the padding of the data within the signature process are generated by using the TSF F.CRYPTO for random number generation. Furthermore, for the signature calculation itself, the TSF makes use of the TSF F.CRYPTO, and the computation of hash values is as well based on the TSF F.CRYPTO. Each private key used for the signature generation function is either generated on-card by usage of the TSF SS.RSA_KeyGen or is generated by the external world and loaded onto the card during the initialisation, personalisation or end-usage phase of the TOE. In the latter case, it is in the responsibility of the external world to guarantee for a sufficient cryptographic strength of the private key and to handle the private key outside the card in a sufficient secure manner. The resistance of the TSF against SPA, DPA, DFA and TA is part of the TSFs SF.Log and F.SIDE_CHAN. For each private key - generated on-card or imported with the assumption that the external world meets the requirements on the key handling as defined before - the TSF digital signature function works in such a manner that the private key cannot be derived from the signature and the signature cannot be generated by other individuals not possessing that secret. Furthermore, the TSF digital signature function works in such a manner that no information about the private key can be disclosed during the generation of the digital signature.
F.VER_DIGSIG	**RSA Verification of Digital Signatures**
	The TSF provides a functionality to verify digital signatures based on asymmetric cryptography, particularly based on the RSA algorithm with key lengths of 2048 bit modulus length. The TSF function to verify a digital signature will be used for several purposes with different formats for the digital signature input: - Implicit verification of digital signatures within authentication mechanisms for the verification of authentication tokens using the signature scheme with message recovery according to the standard /ISO 9796-2/ based on the hash algorithm SHA-256, /EXS_EHC_1/, Chapters 7 and 16 - Implicit verification of digital signatures within the verification and unwrapping of imported CV certificates using the signature scheme with message recovery according to the standard /ISO 9796-2/ based on the hash algorithm SHA-256, see

	/EXS_EHC_1/, Chapters 7, 8 and 16 The TSF function to verify a digital signature uses the public key which has been referenced before. For the verification mechanism itself, the TSF makes directly use of the TSF F.CRYPTO, and the computation of hash values is as well based on the TSF F.CRYPTO. Each public key used for the function to verify a digital signature is either generated on-card by usage of the TSF SS.RSA_KeyGen or is generated by the external world and loaded onto the card during the initialisation, personalisation or end-usage phase of the TOE. In particular, loading via a CV certificate by a suitable preceding operation is possible.
F.RSA_ENC	**RSA Encryption**
	The TSF provides a functionality to encrypt data based on asymmetric cryptography, particularly based on the RSA algorithm with key lengths of 2048 bit modulus length. The TSF encryption function will be used for several purposes with different formats for the encryption input: - Implicit encryption within authentication mechanisms for the generation of authentication tokens using the “encryption primitive” according to the standard /PKCS1/, Chapter 5.1.1 The TSF encryption function uses the public key which has been referenced before. For the encryption mechanism itself, the TSF makes directly use of the TSF F.CRYPTO. Each public key used for the encryption function is either generated on-card by usage of the TSF SS.RSA_KeyGen or is generated by the external world and loaded onto the card during the initialisation, personalisation or end-usage phase of the TOE. In particular, loading via a CV certificate by a suitable preceding operation is possible.
F.RSA_DEC	**RSA Decryption**
	The TSF provides a functionality to decrypt data based on asymmetric cryptography, particularly based on the RSA algorithm with key lengths of 2048 bit modulus length. The TSF decryption function will be used for several purposes with different formats for the data supplied within the cryptogram: - Explicit decryption of a cryptogram using the “decryption scheme” with formatted input according to the standard /PKCS1/, Chapter 7.2.2 and with hash algorithm SHA-256, see /EXS_EHC_1/, Chapter 7 - Implicit decryption within authentication mechanisms for the verification of authentication tokens using the “decryption primitive” according to the standard /PKCS1/, Chapter 5.1.2 The TSF decryption function uses the private key which has been referenced before. For the decryption mechanism itself, the TSF makes directly use of the TSF F.CRYPTO. Each private key used for the decryption function is either generated on-card by usage of the TSF SS.RSA_KeyGen or is generated by the external world and loaded onto the card during the initialisation, personalisation or end-usage phase of the TOE. In the latter case, it is in the responsibility of the external world to guarantee for a sufficient cryptographic strength of the private key and to handle the private key outside the card in a sufficient secure manner. The resistance of the TSF against SPA, DPA, DFA and TA is part of the TSFs SF.Log and F.SIDE_CHAN. For each private key - generated on-card or imported with the assumption that the external world meets the requirements on the key handling as defined

	before - the TSF decryption function works in such a manner that the private key cannot be derived from the cryptogram and the cryptogram cannot be deciphered by other individuals not possessing that secret. Furthermore, the TSF decryption function works in such a manner that no information about the private key may be disclosed during the decipherment of the cryptogram.

7.2 Statement of compatibility

The following chapter contains a statement of compatibility between the platform security target and this composite security target.

The chapter is not disclosed in the ST-Lite.

Reference

I Bibliography

/CC_P1/
Title: Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model
Identification: CCMB-2012-09-001
Version: Version 3.1 Revision 4
Date: September 2012

/CC_P2/
Title: Common Criteria for Information Technology Security Evaluation, Part 2: Security functional components
Identification: CCMB-2012-09-002
Version: Version 3.1 Revision 4
Date: September 2012

/CC_P3/
Title: Common Criteria for Information Technology Security Evaluation, Part 3: Security assurance components
Identification: CCMB-2012-09-003
Version: Version 3.1 Revision 4
Date: September 2012

/CEM/
Title: Common Methodology for Information Technology Security Evaluation – Evaluation methodology
Identification: CCMB-2012-09-004
Version: Version 3.1 Revision 3
Date: September 2012

/AIS36/
Title: Kompositionsevaluierung
Identification: AIS 36, Version 3
Date: 19.10.2010
Publisher: Bundesamt für Sicherheit in der Informationstechnik

/AIS_2031_RNG/
Title: A proposal for: Functionality classes for random number generators
Identification AIS 20 / 31
Version: 2.0
Author: W. Killmann and W. Schindler
Date: 18. September 2011
Publisher: BSI

/BSI-PP-0035/
Title: Smartcard IC Platform Protection Profile
Identification: Registered and Certified by Bundesamt für Sicherheit in der Informationstechnik (BSI) under the reference BSI-PP-0035
Version: Version 1.0
Date: 15.06.2007
Author: Atmel, Infineon Technologies AG, NXP Semiconductors, Renesas Technology Europe Ltd., STMicroelectronics

/CL_UG/
Title: User Guidance Manual: Crypto Library on SmartMX2 – Preparative procedures and operational user guidance
Version: Revision 1.0
Date: 05th December 2012
Publisher: NXP Semiconductors GmbH

/CL_UG_DES/
Title: User Manual: Crypto Library on the SmartMX2 – DES Library
Version: Revision 1.0
Date: 05th December 2012
Publisher: NXP Semiconductors GmbH

/CL_UG_RSA/
Title: User Manual: Crypto Library on the SmartMX2 – RSA Library
Version: Revision 1.0
Date: 05th December 2012
Publisher: NXP Semiconductors GmbH

/CL_UG_RND/
Title: User Manual: Crypto Library on the SmartMX2 – RNG Library
Version: Revision 1.0
Date: 05th December 2012
Publisher: NXP Semiconductors GmbH

/CL_UG_SHA/
Title: User Manual: Crypto Library on the SmartMX2 – SHA Library
Version: Revision 1.0
Date: 05th December 2012
Publisher: NXP Semiconductors GmbH

/CL_UG_RSAKG/
Title: User Manual: Crypto Library on the SmartMX2 – RSA Key Generation Library
Version: Revision 1.0
Date: 05th December 2012
Publisher: NXP Semiconductors GmbH

/CL_UG_UTL/
Title: User Manual: Crypto Library on the SmartMX2 – Utils Library
Version: Revision 1.0
Date: 05th December 2012
Publisher: NXP Semiconductors GmbH

/ST_IC/
Title: Security Target Lite – P60x080/052/040PVC/PVC(Y)
Identification: BSI-DSZ-CC-0837
Version: 1.2
Date: 18 December 2013
Publisher: NXP Semiconductors GmbH

/ST-IC+CL/
Title: Security Target – Crypto Library V1.0 on the P60x080/052/040PVC(Y) Identification: NSCIB-CC-12-36243
Version: Revision 1.1
Date: 20.02.2014
Publisher: NXP Semiconductors GmbH

/NXP_IC_CL_MA/
Title: Assurance Continuity Maintenance Report – Crypto Library V1.0 on P60x080/052/040PVC/PVC(Y)
Identification: NSCIB-CC-12-36243-MA1
Version: 1.0
Date: 09.04.2014
Publisher: TÜV Rheinland Nederland B.V.

/NXP_IC_MA/
Title: Assurance Continuity Maintenance Report
Identification: BSI-DSZ-CC-0837-2013-MA-01
Version: 1.0
Date: 04.02.2014
Publisher: Bundesamt für Sicherheit in der Informationstechnik (BSI)

/ISO 9796-2/
Title: Information Technology – Security Techniques – Digital Signature Schemes Giving Message Recovery – Part 2: Integer Factorization Based Mechanisms

Identification: ISO/IEC 9796-2
Version: Second Edition
Date: 2002
Publisher: ISO / IEC

/ISO_7816_4/
Title: Identification cards – Integrated circuit cards – Part 4: Organization, security and commands for interchange
Identification: ISO/IEC 7816-4
Version: Second edition
Date: 15th Jan. 2005
Publisher: International Organization for Standardization/International Electrotechnical Commission

/SHA/
Title: Secure Hash Standard (SHS)
Identification: FIPS Publication 180-2
Date: August 2002
Publisher: National Institute of Standards and Technology (NIST)

/FIPS 46-3/
Title: Data Encryption Standard (DES)
Identification: FIPS Publication 46-3
Date: October 1999
Publisher: National Institute of Standards and Technology (NIST)

/ANSI X9.52/
Title: Triple Data Encryption Algorithm Modes of Operation
Identification: ANSI X9.52
Date: 1998
Publisher: American National Standards Institute (ANSI)

/EXS_NIST_SP800_38A/
Title: NIST Special Publication 800-38A, Recommendation for Block, Cipher Modes of Operation, Methods and Techniques
Identification 800-38A 2001 ED
Date: December 2001
Author: Morris Dworkin
Publisher: NIST

/PKCS1/
Title: PKCS #1 v2.1: RSA Cryptography Standard
Date: June 2002
Publisher: RSA Laboratories

/ANSI X9.19/
Title: Financial Institution Retail Message Authentication
Identification: ANSI X9.19

Date: 1996
Publisher: American National Standards Institute (ANSI)

/ANSI X9.63/
Title: Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography
Identification: ANSI X9.63
Date: 2001
Publisher: American National Standards Institute (ANSI)

/EXS_EHC_1/
Title: Spezifikation der elektronischen Gesundheitskarte, Teil 1: Spezifikation der elektrischen Schnittstelle
Version: as specified in the Base Roll-Out Release 0.5.3 eGK V1.0.0 (including SRQ supplements)
Date: 11.04.2011
Publisher: gematik mbH

/EXS_EHC_2/
Title: Die Spezifikation der elektronischen Gesundheitskarte, Teil 2: Grundlegende Applikationen (früher: Anwendungen und anwendungsspezifische Strukturen)
Version: as specified in the Base Roll-Out Release 0.5.3 eGK V1.0.0(including SRQ supplements)
Date: 11.04.2011
Publisher: gematik mbH

/ALGCAT/
Title: Geeignete Algorithmen zur Erfüllung der Anforderungen nach §17 Abs.1 bis 3 SigG vom 22. Mai 2001 in Verbindung mit Anlage 1 Anschnitt I Nr. 2 SigV vom 22. Nov. 2001
Identification: Bundesanzeiger (to be published)
Date: 22.12.2010
Publisher: Bundesnetzagentur

/BSI_PP_EHC/
Title Protection Profile – electronic Health Card (eHC) – elektronische Gesundheitskarte (eGK)
Identification BSI-PP-0020-V3-2010-MA-01
Version 2.90
Date April 19th 2011
Publisher Bundesamt für Sicherheit in der Informationstechnik (BSI)

II Summary of abbreviations

A.x	Assumption
AC	Access Condition
AID	Application Identifier
ALW	Always
AM	Access Mode
AR	Access Rule
AS	Application Software
ATR	Answer To Reset
AUT	Key Based Authentication
BS	Basic Software
CC	Common Criteria
CGA	Certification Generation Application
CH	Card Holder
CHV	Cardholder Verification
CSP	Certification Service Provider
DES	Data Encryption Standard
DF	Dedicated File
DFA	Differential Fault Analysis
DPA	Differential Power Analysis
DTBS	Data to be signed
EAL	Evaluation Assurance Level
EF	Elementary File
EHC	Electronic Health Card
ES	Embedded Software
HPC	Health Professional Card
IC	Integrated Circuit
IFD	Interface Device
MAC	Message Authentication Code
MF	Master File
O.x	Security Objective
OS	Operating System
PAR	Partial Access Rule
P.x	Organisational Security Policy
PIN	Personal Identification Number
PP	Protection Profile
PUC	PIN Unblocking Code
PW	Password
PWD	Password Based Authentication
RAD	Reference Authentication Data
RSA	Rivest-Shamir-Adleman Algorithm
SAR	Security Assurance Requirement
SCA	Signature Creation Application
SCD	Signature Creation Data
SCS	Signature Creation System
SDO	Signed Data Object
SFP	Security Function Policy
SFR	Security Functional Requirement
SM	Secure Messaging
SMC	Security Module Card
SOF	Strength of Functions

SPA	Simple Power Analysis
SPM	TOE Security Policy Model
SSC	Send Sequence Counter
SSCD	Secure Signature Creation Device
ST	Security Target
SVD	Signature Verification Data
TA	Timing Analysis
T.x	Threat
TOE	Target of Evaluation
TSC	TSF Scope of Control
TSF	TOE Security Function
TSP	TOE Security Policy
VAD	Verification Authentication Data

III Glossary

For explanation of technical terms refer to the following documents:

/BSI-PP-0035/, Chap. 8.7