banksys
end-to-end transactions

Haachtsesteenweg
1442
1130 Brussels
Belgium

Terminals & Card Applications

DEP/PCI

Security Target

Document version: 1.1 (4)
Document date: 08 May 2003

TABLE OF CONTENTS

Table of Contents ***3***
1. Security Target Introduction ***5***
1.1. ST IDENTIFICATION 5
1.2. ST OVERVIEW 5
1.3. CC CONFORMANCE 6
2. TOE Description ***7***
2.1. OVERVIEW 7
2.2. PHYSICAL BOUNDARIES OF THE TOE 8
2.3. LOGICAL BOUNDARIES OF THE TOE 10
2.3.1. Software parts *10*
2.3.2. Software Interfaces *12*
2.4. TOE BOUNDARIES SUMMARY 13
2.5. TOE MANAGEMENT AND AUTHORITY LEVELS 15
2.5.1. Administrators *15*
2.5.2. NONE Authority Level *16*
2.5.3. INIT Authority Level *17*
2.5.4. BKS Authority Level *17*
2.5.5. CUST Authority Level *17*
2.5.6. Allowed Authority Level Changes *18*
3. TOE Security Environment ***19***
3.1. DEFINITION OF SUBJECTS, OBJECTS AND OPERATIONS 19
3.1.1. Definition of Subjects: *19*
3.1.2. Definition of Data Objects *20*
3.1.3. Definition of Operations *20*
3.1.4. Security attributes of operations *21*
3.2. ASSUMPTIONS 22
3.3. THREATS 23
3.4. ORGANISATIONAL SECURITY POLICIES 24
4. Security Objectives ***26***
4.1. TOE SECURITY OBJECTIVES 26
4.1.1. TOE Security Objectives in all Authority Levels: *26*
4.1.2. TOE Security Objectives in CUST Authority Level *27*
4.2. SECURITY OBJECTIVES FOR THE ENVIRONMENT 29
5. IT Security Requirements ***31***
5.1. TOE SECURITY FUNCTIONAL REQUIREMENTS 31
5.1.1. Services offered by the TOE *32*
5.1.2. Loading Application Software *37*
5.1.3. Loading, backing up, and deleting Application Keys *39*
5.1.4. Management of the TOE *41*
5.1.5. Tampering and abnormal operating conditions *45*
5.1.6. Architectural security *48*
5.1.7. Strength-of-function claim *49*
5.2. TOE SECURITY ASSURANCE REQUIREMENTS 50

5.3. SECURITY REQUIREMENTS FOR THE IT ENVIRONMENT 51
5.4. EXPLICITLY STATED REQUIREMENTS 53
5.4.1. FMT_SAE.2 Use-limited authorization 53
5.4.2. FCS_RND.1 Quality metrics for random numbers. 54
6. TOE Summary Specification 55
6.1. TOE SECURITY FUNCTIONS 55
6.1.1. Cryptographic service functions 55
6.1.2. Loading and saving security functions 56
6.1.3. Managing the TOE 57
6.1.4. Tampering/Abnormal conditions security functions 58
6.1.5. Architectural security 58
6.1.6. Probabilistic functions and mechanisms 58
6.2. ASSURANCE MEASURES 59
7. PP Claims 62
8. Rationale 63
8.1. SECURITY OBJECTIVES RATIONALE 63
8.2. SECURITY REQUIREMENTS RATIONALE 66
8.2.1. The SFRS meet the Security Objectives for the TOE 66
8.2.2. The security requirements for the IT environment meet the security objectives for the environment 67
8.2.3. The Assurance Requirements and Strength of Function Claim are appropriate 68
8.2.4. All dependencies have been met 68
8.2.5. The requirements are internally consistent 69
8.2.6. The requirements are mutually supportive 70
8.3. TOE SUMMARY SPECIFICATION RATIONALE 71
8.3.1. The functions meets the SFRs 71
8.3.2. The assurance measures meets the SARs 74
8.3.3. The SOF-claims for functions meet the SOF-claims for the SFRs 74
8.3.4. The functions are mutually supportive 74
8.4. PP CLAIMS RATIONALE 74
9. Annexes 75
9.1. GLOSSARY 75
9.2. REFERENCES 76
9.3. FIPS 140-2 TESTS 77
9.4. DOCUMENT HISTORY 78

1. SECURITY TARGET INTRODUCTION

1.1. ST IDENTIFICATION

Name of the TOE: banksys DEP/PCI version 3.0

Name of the Security Target: banksys DEP/PCI Security Target, version **1.1 (4)**

ST evaluation status: Final evaluated version

1.2. ST OVERVIEW

The "Data Encryption Peripheral PCI" (called DEP/PCI in the following) is a Host Security Module (HSM) that can for example be used in banking, government, pay-TV and e-commerce environments.

It consists of hardware and software and provides the following functionality:

- Loading of data: e.g. an application and application keys.
- Execution of cryptographic operations like DES, Triple-DES, AES, RSA, CBC-MAC computation, hashing, digital signature computation, key generation, random generation.

The confidentiality and integrity of all data in the DEP/PCI is protected:

- Physically, by tamper resistance and tamper responsive hardware,
- Logically, by only allowing well defined interfaces and using access control (permissions to execute a specific task).

1.3. CC CONFORMANCE

The evaluation is based upon:

- Common Criteria for Information Technology Security Evaluation, Version 2.1, Part 1: General model, August 1999.
- Common Criteria for Information Technology Security Evaluation, Version 2.1, Part 2: Security functional requirements, August 1999.
- Common Criteria for Information Technology Security Evaluation, Version 2.1, Part 3: Security assurance requirements, August 1999.
- Common Methodology for Information Technology Security Evaluation, Version 1.0, Part 2: Evaluation Methodology, August 1999.

In addition, the following interpretations to these criteria and methodology were used:

- All CCRA Final Interpretations up to and including February 28, 2002
- BSI Interpretations: AIS 1 v7, AIS 14 v1, AIS 19 v1, AIS 31 v1, AIS 32 v1

The chosen level of assurance is:

EAL3 (Evaluation Assurance Level 3) + ADV_FSP.2

This Security Target claims the following conformances:

CC Part 2 extended
CC Part 3 conformant

no conformance to any PP.

2. TOE DESCRIPTION

2.1. OVERVIEW

The TOE is the banksys product DEP/PCI, version 3.0. It is a tamper-resistant and tamper-responsive host security module, which can be used with standard PC hardware that supports a PCI interface.

The DEP/PCI is a generic platform providing cryptographic services, e.g. DES, Triple-DES (3DES), AES, RSA, CBC-MAC computation, hashing, digital signature computation, key generation, random generation. It is meant to provide security services required by different application domains like EFT, Electronic Purse, e-commerce, PKI, etc.

The main use of the TOE is at the host side (e.g. it is plugged into a workstation that is connected to a mainframe or server located in a computer room, or it is plugged into a server located in a computer room).

The TOE provides means to securely load an application and keys into it. Only authorised personnel (e.g. a security officer) can enable the loading of applications and/or keys.

The DEP/PCI includes hardware (e.g. a main processor board, an alarm processor board and cryptographic co-processors) and software (e.g. standard libraries) components. The TOE communicates with its environment via a PCI-bus and serial ports.

The DEP/PCI detects tamper attacks (e.g. intrusion, temperature and chemical attacks) and takes appropriate measures to log the event and to protect all sensitive data.

2.2. PHYSICAL BOUNDARIES OF THE TOE

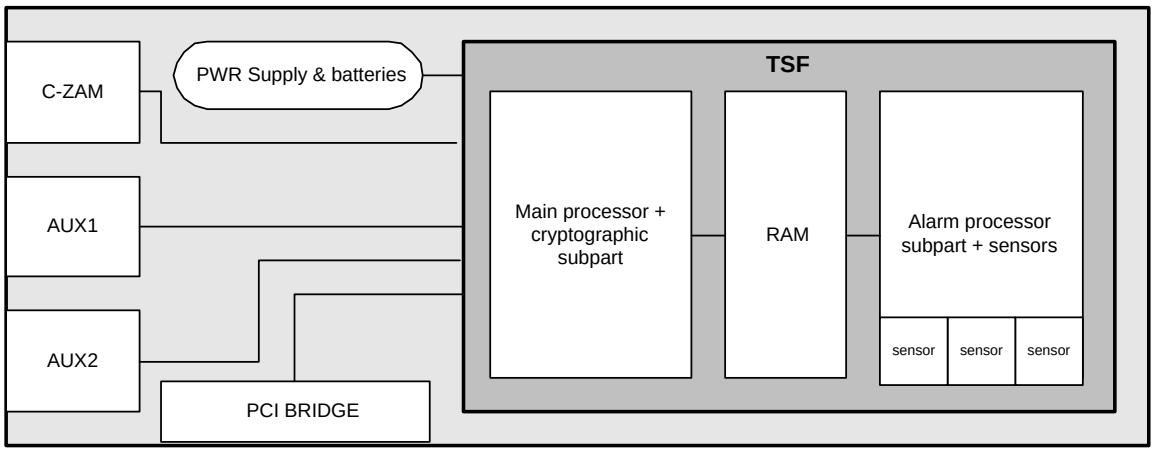


Figure 1: TOE-Hardware

Physically the TOE is a PCI-card that can be plugged into any workstation that supports PCI cards.

Two main parts of the TOE can be distinguished:

- a secured module (the TSF), containing mainly:
  - the main processor and cryptographic co-processors,
  - an alarm processor,
  - RAM that can be accessed by both processors,
  - alarm sensors.
- the PCI module, containing mainly:
  - power supply and batteries
  - the PCI bridge,
  - serial line 'C-ZAM': to connect a C-ZAM/DEP (an external chip card encoder/reader that is used for administrative purposes),
  - serial line 'AUX1': is not used[1],

[1] See section 2.3 for a more detailed description.

- serial line 'AUX2': to communicate with the alarm board of the TOE (e.g. logging security incidents to a printer or another device).

The TOE is used in the following hardware environment:

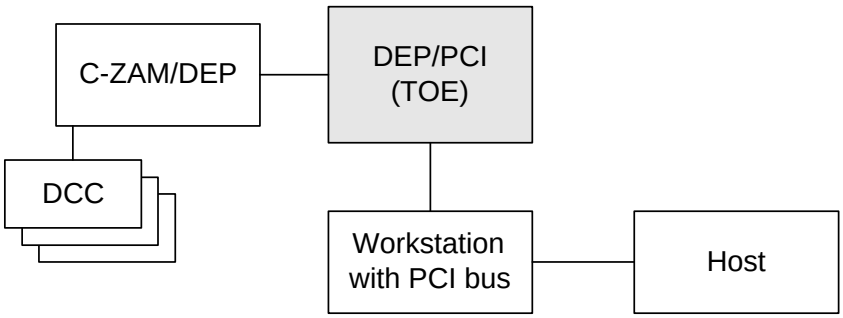


Figure 2: TOE-Hardware Environment

The Host uses cryptographic services provided by the TOE. To do so it is connected to a workstation containing the DEP/PCI.

The C-ZAM/DEP is a chip card encoder/reader with its own keyboard and display. It is used for administration purposes and uses smart cards (called DCCs: DEP Control Cards) to store information relevant for securely administering the TOE. The C-ZAM/DEP itself is not part of the TOE and falls outside the scope of the evaluation.

2.3. LOGICAL BOUNDARIES OF THE TOE

The logical boundaries of the TOE are shown in Figure 3:

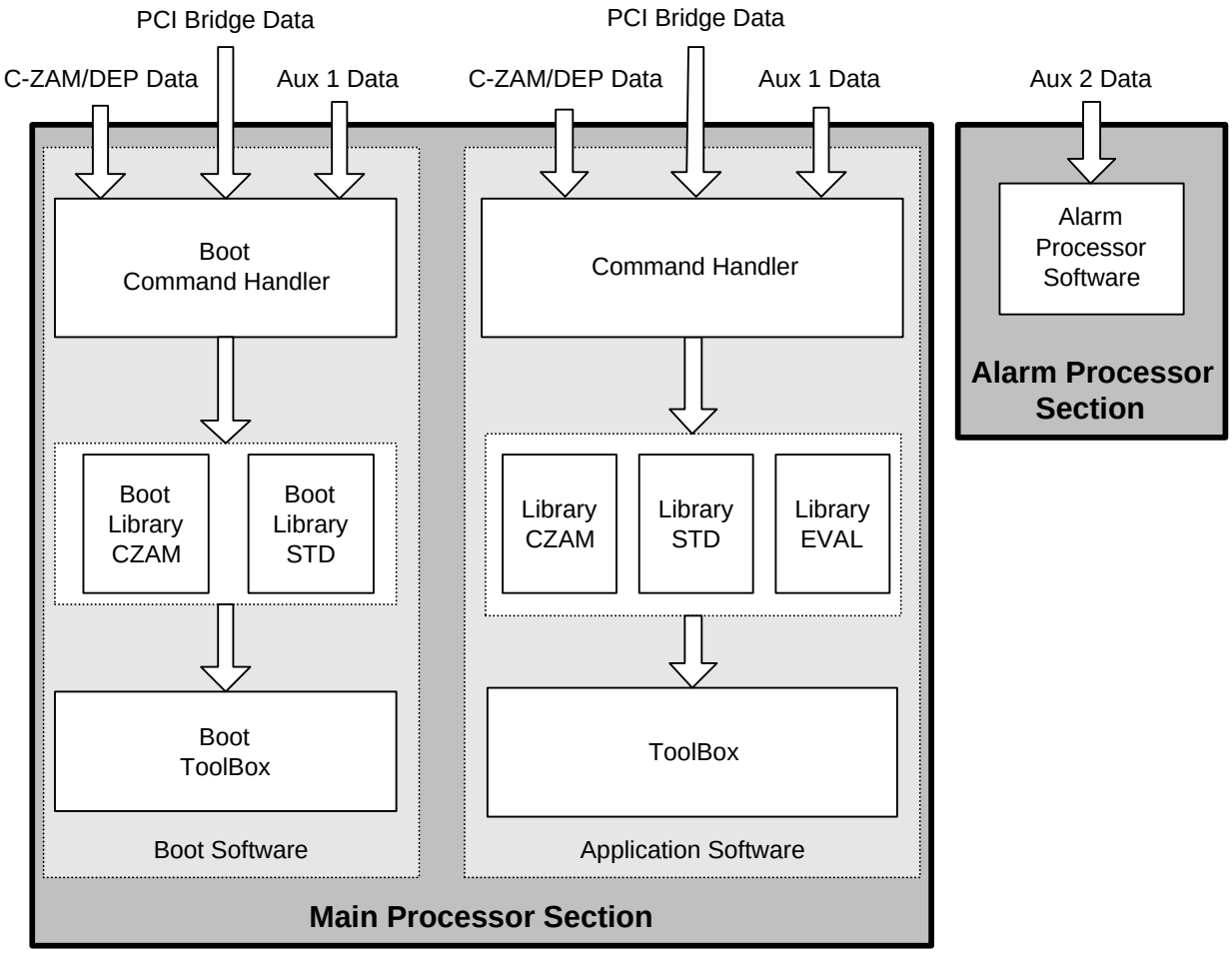


Figure 3: TOE-Software Environment

2.3.1. Software parts

The DEP/PCI contains three major software parts:

- the *Boot Software* (in EEPROM). This part executes while no Application Software has been loaded. As soon as Application Software is loaded, this part is "switched off" and execution is transferred to the Application Software;
- the *Application Software* (in RAM). This part executes when it is loaded, and provides the operational cryptographic services of the TOE;
- the *Alarm Processor Software* (in EEPROM). This part executes concurrently with the other two, and continuously monitors the various sensors of the TOE for alarms. If an alarm is triggered, it removes the Application Software, and transfers back control to the Boot Software, effectively resetting the TOE completely.

The first two of these (Boot Software and Application Software) are detailed below.

The *Boot Software* contains the following major parts:

- the **Boot Command Handler** contains the communication protocols of the different ports. It forwards incoming messages to the correct Boot Library,
- the **Boot Library CZAM** contains the functionality needed by the C-ZAM/DEP interface:
  - authentication of the DEP/PCI Main Processor Section hardware,
  - defining the DEP/PCI Customer, Use Mode, and Authority Level,
  - loading keys in the DEP/PCI,
  - loading capabilities in the DEP/PCI.
- the **Boot Library STD** contains the functionality needed by the PCI Bridge interface:
  - loading Application Software,
  - testing: inquiring DEP/PCI status, communication testing, self-test, internal diagnostics,
- the **Boot ToolBox** contains cryptographic functions used by the Boot Libraries.

The *Application Software* contains the following major parts:

- the **Command Handler** contains the communication protocols of the different ports. It executes pre-processing (checks on incoming data formats) for incoming PCI Bridge messages. It parses incoming messages into elementary interfaces and calls the correct Libraries,
- the **Library CZAM** contains the functionality needed by the C-ZAM/DEP interface (same functionality as the Boot Library CZAM):
  - authentication of the DEP/PCI Main Processor Section hardware,
  - defining the DEP/PCI use mode, customer and authority mode,
  - loading keys in the DEP/PCI,
  - loading capabilities in the DEP/PCI,
- the **Library STD** contains functionality needed by the PCI Bridge interface:
  - deleting Application Software,
  - backup and restore of Application Keys,
  - defining and reading parameter values,
  - testing: inquiring DEP/PCI status, communication testing, self-test, internal diagnostics,
  - management: inquiring for software version, inquiring lists of available libraries and interfaces, inquiring counting information (key and capability loading, interface execution, the number of times errors occurred),
- the **Library EVAL** contains an example of cryptographic operations that can be executed by the DEP/PCI. This Library was specifically designed[2] for the CC-

[2] banksys offers a variety of other libraries to customers. Examples of libraries are: PKI, EMV and customer specific libraries. These libraries are not part of the evaluation.

evaluation to showcase the cryptographic and random number generation functionality of the DEP/PCI.
- the **ToolBox** contains cryptographic functions used by the Libraries.

2.3.2. Software Interfaces

The following software interfaces exist in the TOE:

PCI-bridge interface
This is used for communication (through the Command Handler) with the Boot Library STD, the Library STD and the Library EVAL.

CZAM/DEP interface
This is used for communication (through the Command Handler) with the CZAM Boot Library and the CZAM library.

AUX 1 is not used**.**
The Library EVAL does not use the AUX1 interface. It may be the case that other libraries (see footnote 2) will use this interface, but these libraries are not part of the evaluation. “Not used” means that the TOE does not send data to AUX1, and the TOE ignores all incoming data from AUX1.

AUX 2 is used for:
- authentication of the DEP/PCI Alarm Processor Section hardware,
- the reading alarm status and alarm logging,
- alarm processor administration purposes.

2.4. TOE BOUNDARIES SUMMARY

The following lists summarise the TOE-components:

TOE IT components	
Hardware	DEP/PCI
Software	Alarm Processor Software
	Boot Software: Boot Command Handler, Boot Library CZAM, Boot Library STD, Boot ToolBox
	Application Software OS: Command Handler, Library CZAM, Library STD, ToolBox
	Application Software extra library: EVAL Library[3]
External interfaces	PCI bridge protocol
	C-ZAM/DEP protocol
	Sensor interfaces
	AUX 2 (Alarm) protocol
	AUX 1 protocol

TOE Guidance components	
Administrator guidance	• DEP/NT Documentation – DEP/NT Installation Guide • DEP/PCI – Customer Security Officer: Guidelines • DEP/PCI – Customer Host Programmers Guidelines • DEP/NT Documentation – DEP/NT C-ZAM/DEP User Manual • DEP/NT Documentation – DEP/NT PC-AUX Program User Manual • DEP/NT Documentation – DEP/NT Host Interface Supervision User Manual • DEP/NT Documentation – DEP/NT DEP Handler Supervision User Manual

[3] This part of the Application Software was created especially for the evaluation: it is designed to showcase all functionality of the TOE.

TOE Guidance components	
Evaluated version specific user guidance	• Common Criteria Software – Integration Manual • Subset of Eval Library for DEP – Reference DFS Manual • DEP/PCI – Security Target • DEP/PCI – Guidance Documentation – Erratum
General user guidance	• Subset STD Library for DEP – Ref DFS Manual

2.5. TOE MANAGEMENT AND AUTHORITY LEVELS

2.5.1. Administrators

The Authority Level of the DEP/PCI is the state the device is currently in. The DEP/PCI can only be in one state at the same time. The Authority Level defines which type of administrator is allowed to perform administrative actions.

A DEP is always under **one** of the following Authorities:

- No Authority (**NONE**),
- Initial Authority (**INIT**) under the control of the INIT Administrator
- Banksys Authority (**BKS**) under the control of the BKS Administrator
- Customer Authority (**CUST**) under the control of the CUST Administrator

The Authority Level concept was introduced because it is important that not everybody can manage the TOE, because this would allow logical access to the TOE to perform security-critical actions. For this reason, an authentication mechanism was built into the TOE, and three classes of administrator were defined:

- INIT administrator
- BKS administrator
- CUST administrator

Each administrator can authenticate himself to the TOE with a C-ZAM/DEP containing a secret unique for that administrator. This secret is then used to create a trusted channel (protection against modification and disclosure) between the C-ZAM/DEP of that administrator and the TOE. This trusted channel is subsequently used for all relevant management actions.

The INIT administrator (normally banksys staff) uses the INIT secret which is hardcoded into the TOE and identical for every TOE. This means that this secret is (or can be known) by banksys developers (both of the TOE and the C-ZAM/DEP), people with access to the TOE code, the manufacturer of the TOE and the banksys security officers. Therefore this secret is insufficiently secret to use it for operational administration of the TOE.
During installation the BKS administrator (normally banksys staff) therefore defines a new secret, unique to that particular TOE. Only the BKS administrator knows this secret. However, customers prefer to use secrets only they know, so a third administrator, the CUST administrator (normally an employee of the customer) defines a third secret. This secret is subsequently used for operational administration.

The different Authority Levels and actions that administrators can undertake are described more in detail in the next paragraphs.

2.5.2. NONE Authority Level

The None Authority level is obtained just after the manufacturing phase and after a complete reset (with the deletion of the complete contents of the RAM) of the DEP/PCI. Whenever a DEP/PCI is first switched on (or after an alarm has been generated), the DEP/PCI is in the NONE Authority Level, with no customer assigned, and no mode set (see below).

In NONE Authority, three actions can be taken:

1. Assign the DEP/PCI to a specific customer. Once assigned, this customer can only be changed by completely resetting the TOE.
2. Set the DEP/PCI to a specific Use Mode. The TOE distinguishes three different use modes[4]:
   - **DEV** – Development mode, used by the developer of DEP Applications;
   - **TST** – Test mode, used for the testing of the DEP and DEP Applications;
   - **LIV** – Live mode, used for real productive operation.

   Once set, this Use Mode can only be changed by completely resetting the DEP/PCI.
3. Raise the Authority Level of the DEP/PCI to INIT (only after use mode and customer have been set)

Note: If the DEP/PCI is reset while in NONE Authority level or detects a tamper attempt it stays on NONE Authority level, but removes the customer (if assigned) and the Use Mode (if set).

The DEV, TST and LIV mode only differ in their choice of INIT secrets (see the following section). The TOE contains three sets of INIT secrets: one for each mode. All other functionality of the TOE is completely identical. The setting of a mode selects which of three secrets to use. Banksys wishes to restrict knowledge of the LIV mode INIT secrets as much as possible. Banksys carries out all of its own testing in TST mode, including the developer testing done during this evaluation.

The entire evaluation has therefore been carried out in the TST mode.

[4] See the boxed section below for more information on these modes.

2.5.3. INIT Authority Level

Once the DEP/PCI is in INIT level, only the INIT administrator (in possession of a C-ZAM/DEP with INIT secrets) can use the DEP/PCI. The INIT secrets are hardcoded in every DEP allowing a trusted channel between C-ZAM/DEP and DEP/PCI.

This INIT administrator controls the DEP/PCI until it is provided to the BKS administrator. In INIT level, only one action can be taken:

1. Raise the Authority Level of the DEP/PCI to BKS by loading a BKS secret into it.

Note: If the DEP/PCI is reset while in INIT Authority level or detects a tamper attempt it reverts to NONE Authority level and removes the customer and the Use Mode.

2.5.4. BKS Authority Level

Once the DEP/PCI is in BKS level, only the BKS administrator (in possession of a C-ZAM DEP with BKS secrets) can use the DEP/PCI. The BKS secrets were loaded in the DEP earlier, and allow a trusted channel between C-ZAM/DEP and DEP/PCI.

This BKS administrator (normally a Banksys employee) controls the DEP/PCI until the DEP/PCI is provided to the CUST Administrator. In BKS level, only one action can be taken:

1. Raise the Authority Level of the DEP/PCI to CUST by loading CUST secrets into it.

Note: If the DEP/PCI is reset while in BKS Authority level or detects a tamper attempt it reverts to NONE Authority level and removes the customer, the Use Mode and the BKS secret.

2.5.5. CUST Authority Level

Once the DEP/PCI is in CUST level, two types of actions can be undertaken:

- Management actions
- Operational actions

Management actions

Only the CUST administrator (in possession of a C-ZAM DEP with CUST secrets) can perform these actions. The CUST secrets were loaded in the DEP earlier, and allow a trusted channel between C-ZAM/DEP and DEP/PCI. The following actions can be undertaken:

- Enabling the actions 1-4 below so that they can be used. The actions can be enabled either indefinitely, or for a limited time only, or for a limited number of uses
- Disabling the actions 1-4 below, so that they cannot be used anymore

Operational Actions
These can be undertaken by anyone having logical access to the PCI-bridge interface:

1. Loading Application Software (if enabled). Application Software has to be signed by the BKS administrator otherwise it will be rejected
2. Loading Application Keys (if enabled)
3. Backing-up/Restoring Application Keys (if enabled)
4. Providing Random Number Generation (if enabled)
5. Providing Cryptographic services by the Application Software (DES, Triple-DES (3DES), AES, RSA, CBC-MAC computation, hashing, digital signature computation, key generation

Note: If the DEP/PCI is reset while in CUST Authority level or detects a tamper attempt it reverts to NONE Authority level and removes the customer, the Use Mode, the BKS secret, the CUST secret, the Application Software and the Application Keys.

2.5.6. Allowed Authority Level Changes

The allowed changes between Authority Levels are depicted in Figure 4.

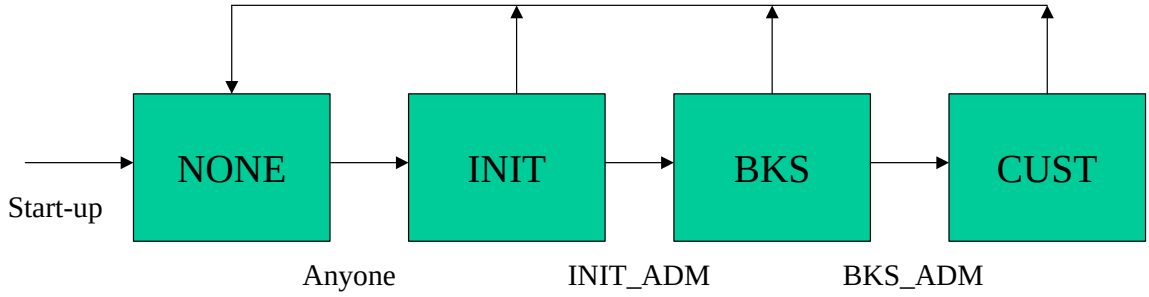


Figure 4: Allowed Authority Level Changes

3. TOE SECURITY ENVIRONMENT

3.1. DEFINITION OF SUBJECTS, OBJECTS AND OPERATIONS

To facilitate easy definition of threats, OSPs, assumptions, security objectives and security requirements, we define the subjects, objects and operations to be used in the ST first.

3.1.1. Definition of Subjects:

S.TA_NETWORK	An unauthorised person or process that has a logical connection to either the PCI-bus on which the TOE resides or to the connection between the Workstation on which the TOE resides and the Host.
S.TA_PHYSICAL	An unauthorised person that has physical access to the TOE[5]
S.INIT_ADM	A user (or subject acting on behalf of that user) with the role of INIT administrator.
S.BKS_ADM	A user (or subject acting on behalf of that user) with the role of BANKSYS administrator.
S.CUST_ADM	A user (or subject acting on behalf of that user) with the role of CUSTOMER administrator.

Note that S.TA_PHYSICAL and/or S.TA_NETWORK can be the same person as S.INIT_ADM, S.BKS_ADM and/or S.CUST_ADM.

S.DEP_APPL	Application Software residing upon the TOE. It is the counterpart to S.HOST_APPL and provides services for it by using the TOE functionality. Note that some operations treat S.DEP_APPL as an object.
S.HOST_APPL	Application Software residing on a Host. This software wants to obtain security services from S.DEP_APPL.

[5] As S.TA_PHYSICAL also has access to the PCI-bus, S.TA_PHYSICAL includes S.TA_NETWORK

3.1.2. Definition of Data Objects

D.DEP_APPL	An application that can be loaded into the TOE. After D.DEP_APPL is checked and installed by the TOE it becomes the subject S.DEP_APPL. D.DEP_APPL has the identity of its author as security attribute.
D.APPL_KEYS	Cryptographic application keys that are used by S.DEP_APPL.
D.BKUPD_KEYS	D.APPL_KEYS that have been backed up outside the TOE. Nobody except the TOE can read D.APPL_KEYS.
D.MESSAGE	Commands sent to the TOE by S.HOST_APPL. These commands are requests to S.DEP_APPL to use an operation (see below) on certain D.APPL_KEYS. These keys may be sent as part of D.MESSAGE or it may already reside in the TOE.
S.DEP_APPL	This subject is also an object. It is defined in the list of subjects.

3.1.3. Definition of Operations

R.SERV_EXT	One of the services provided by the TOE to S.HOST_APPL: R.KEYGEN: Key generation for AES, DES, 3DES, RSA R.CRYPT: Encryption/decryption with AES, (3)DES, and RSA R.MAC: CBC-MAC generation with DES, 3DES, AES R.SIGN: Signature generation/verification with RSA R.HASH: Hash generation with SHA-1, SHA-256, MD5 R.RND: Strong random number generation
R.LOAD_APPL	An operation of the TOE that allows it to load D.DEP_APPL and transform it into S.DEP_APPL.
R.ERASE_APPL	An operation of the TOE that allows it to remove S.DEP_APPL and all D.APPL_KEYS.
R.LOAD_KEYS	An operation of the TOE that allows it to load some D.APPL_KEYS.
R.BACKUP_KEYS	An operation of the TOE that allows it to backup some D.APPL_KEYS. This creates the object D.BKUPD_KEYS.
R.RESTORE_KEYS	An operation of the TOE that allows it to restore D.BKUPD_KEYS into D.APPL_KEYS.

R.ERASE_KEYS An operation of the TOE that allows it to delete some D.APPL_KEYS.

3.1.4. Security attributes of operations

R.LOAD_APPL, R.ERASE_APPL, R_LOAD_KEYS, R.BACKUP_KEYS, R_RESTORE_KEYS, R.ERASE_KEYS, and R.RND[6] have a Capability as security attribute, which is either LOADED (the operation is allowed) or UNLOADED (the operation is not allowed).

A Capability can also be:

- Limited in time: it is LOADED until a certain time has elapsed after which it becomes UNLOADED;
- Limited in use: it is LOADED until it has been used a certain number of times after which it becomes UNLOADED

In the rest of the ST the following terms are used to make the ST more readable.

- To *enable* an operation: to set its Capability to LOADED;
- To *disable* an operation: to set its Capability to UNLOADED;
- To *limit* an operation: to limit it either in time or in use.

[6] In theory, any of the services in R.SERV_EXT can have a Capability, as the writer of the Application Software determines this. In this particular Application Software (with the EVAL library instead of some other library) it was chosen to provide only R.RND with a Capability security attribute.

3.2. ASSUMPTIONS

A.ADMIN
The TOE shall be administered in a secure manner. This includes:

- Only trustworthy personnel can fulfil the roles of S.INIT_ADM, S.BKS_ADM, and S.CUST_ADM. This personnel is adequately trained, and keeps their own confidential information, such as passwords, keys, PINs secret)
- A C-ZAM/DEP in INIT mode, a C-ZAM/DEP in BKS mode and a C-ZAM/DEP in CUST mode shall be available. As these serve as the means to authenticate S.INIT_ADM, S.BKS_ADM, and S.CUST_ADM these shall be kept secure by S.INIT_ADM, S.BKS_ADM, and S.CUST_ADM respectively.

Note: This assumption does not indicate that administrative personnel are fully trusted.

A.ENABLE_PROTECT
Whenever an operation is enabled (see P.ADMIN and P.SERVICES_EVAL), anyone with logical access to the TOE can perform that operation. The environment must therefore ensure that only authorised use is made of that operation. This means that S.CUST_ADM should only enable some[7] operations in an environment where unauthorised physical access and unauthorised logical access to the PCI-bridge are impossible.

A.SIGN_SOFTWARE
S.BKS_ADM ensures that the Application Software D.DEP_APPL:

- is correct
- has the right Capabilities LOADED/UNLOADED
- and suitably protects access to all cryptographic keys before signing it.

A.SECURE_ROOM
The TOE will be deployed in a server or workstation in a "server-room" environment that restricts physical access to only necessary personnel[8]. The physical security of the room will be similar to a typical banking/financial institution computer server room. The environmental conditions will be similar to a typical computer server room.

A.KEY_GEN
Any keys generated outside the TOE that are subsequently loaded in the TOE are generated in a confidential way, be unique with a very high probability and cryptographically strong.

[7] Which operations depends on the context in which the TOE is used.

[8] Ideally restricted to only S.CUST_ADM, but in practice other personnel (other sysadmins, security guards, cleaning staff) will also enter.

3.3. THREATS

T.UNAUT_APPLICATION_LOAD
S.TA_PHYSICAL tries to load or replace D.DEP_APPL.

T.UNAUT_KEY_LOAD
S.TA_PHYSICAL tries to load or replace D.APPL_KEYS.

T.UNAUT_KEY_BACKUP
S.TA_PHYSICAL tries to backup or restore D.APPL_KEYS.

T.UNAUT_BACKUP_DISCLOSURE
S.TA_PHYSICAL tries to read D.BKUPD_KEYS.

T.UNAUT_ERASE
S.TA_PHYSICAL tries to erase D.APPL_KEYS and/or S.DEP_APPL from the TOE without resetting the TOE.

T.PHYSICAL_TAMPER
S.TA_PHYSICAL tries to modify the TOE or S.DEP_APPL, or read out D.APPL_KEYS directly from the TOE by physically tampering with the TOE.

T.LOGICAL_TAMPER
S.TA_PHYSICAL tries to modify the TOE or S.DEP_APPL, or read out D.APPL_KEYS directly from the TOE by logically tampering with the TOE.

T.BAD_RANDOM
S.TA_PHYSICAL tries to predict information on random numbers and/or keys generated by the TOE, based on either a poor random number generator, or a random number generator that degrades in time.

T.EAVESDROP
S.TA_NETWORK eavesdrops on the traffic between S.HOST_APPL and S.DEP_APPL.

T.IMPERSONATE
S.TA_PHYSICAL tries to impersonate as S.INIT_ADM, S.BKS_ADM or S.CUST_ADM. Alternatively S.INIT_ADM, S.BKS_ADM or S.CUST_ADM try to impersonate one another.

3.4. ORGANISATIONAL SECURITY POLICIES

P.SERVICES_EVAL
S.HOST_APPL can obtain the following services from the TOE:

- AES key generation, encryption, decryption, and CBC-MAC generation
- DES/3DES key generation, encryption, decryption, and CBC-MAC generation
- RSA key generation, encryption, decryption, signature creation, signature verification
- SHA-1, SHA-256, and MD5 hashing
- Strong random number generation (if enabled)

Those services that use keys can use either keys already loaded in the TOE, or keys to be supplied by S.HOST_APPL. Those services that generate keys can either store them in the TOE or supply them to S.HOST_APPL.
Strong random number generation can be enabled, disabled, limited in time, or limited in the amount of times that S.HOST_APPL can use it.

P.AUTHORITY_LEVELS[9]
The TOE is always in exactly one of the following four Authority Levels:

- NONE: the initial state when the TOE is first turned on, can be set in LIV/TST/DEV Use Mode, and assigned to a customer;
- INIT: a state where the TOE is initialised;
- BKS: a state where the TOE is further personalised
- CUST: the operational state

P.AUTHORITY_LEVEL_CHANGE

- Anyone with physical access to the TOE can move the TOE to NONE Authority Level from any other Authority Level[10], but moving the TOE to NONE completely resets the TOE by removing S.DEP_APPL and D.APPL_KEYS from the TOE, and resetting the customer and Use Mode.
- Anyone with logical access to the TOE can move the TOE from NONE to INIT Authority Level
- Only S.INIT_ADM can move the TOE from INIT to BKS Authority Level
- Only S.BKS_ADM can move the TOE from BKS to CUST Authority Level

[9] See section 2.5 for a more detailed explanation of the reason for this and the next policy.

[10] By triggering an alarm, e.g. by tapping on it.

P.ADMIN

In CUST Authority Level, the TOE can be managed as follows:

- D.DEP_APPL (Application Software) can be loaded, and transformed into S.DEP_APPL
- D.APPL_KEYS can be loaded by S.CUST_ADM from a C-ZAM/DEP
- D.APPL_KEYS can be backed up (into D.BKUPED_KEYS)
- D.BKUPED_KEYS can be restored (into D.APPL_KEYS)
- S.DEP_APPL and D.APPL_KEYS can be deleted from the TOE (without resetting it)

S.CUST_ADM can enable, disable, limit in time or limit in amount of times each of these management functions.

4. SECURITY OBJECTIVES

4.1. TOE SECURITY OBJECTIVES

4.1.1. TOE Security Objectives in all Authority Levels:

O.AUTHORITY_LEVEL
The TOE is always in exactly one of the following four Authority Levels:

- NONE: the initial state when the TOE is first turned on, and can be set in LIV/TST/DEV mode, and assigned to a customer;
- INIT: a state where the TOE is initialised;
- BKS: a state where the TOE is further personalised
- CUST: the operational state

O.AUTHORITY_LEVEL_CHANGE
Anyone with physical access to the TOE can move the TOE to NONE Authority Level from any other Authority Level[11], but moving the TOE to NONE completely resets the TOE by removing S.DEP_APPL and D.APPL_KEYS from the TOE, and resetting the customer and Use Mode.

- Anyone with logical access to the TOE can move the TOE from NONE to INIT Authority Level
- Only S.INIT_ADM can move the TOE from INIT to BKS Authority Level
- Only S.BKS_ADM can move the TOE from BKS to CUST Authority Level

O.ADMIN_I&A
Administrators are identified and authenticated as follows:

- S.INIT_ADM by having a C-ZAM/DEP in INIT level (with the INIT secret in it);
- S.BKS_ADM by having a C-ZAM/DEP in BKS level (with a BKS secret in it);
- S.CUST_ADM by having a C-ZAM/DEP in CUST level (with a CUST secret in it).

O.PHYSICAL_TAMPER
The TOE shall detect physical tampering, log this and delete the S.DEP_APPL and D.APPL_KEYS. Physical tampering includes: penetrating the cover by physical or chemical terms, removing the cover, removing the TOE from the PCI slot, unusual temperature, unusual voltages, and unusual physical acceleration of the TOE.

O.LOGICAL_TAMPER
It shall be impossible to logically tamper with or bypass the TOE security functions.

[11] By triggering an alarm, e.g. by tapping on it.

4.1.2. TOE Security Objectives in CUST Authority Level

O.SERVICES
Once S.DEP_APPL is loaded, the TOE shall provide the following services to S.HOST_APPL:

- AES key generation, encryption, decryption, and CBC-MAC generation
- DES/3DES key generation, encryption, decryption, and CBC-MAC generation
- RSA key generation, encryption, decryption, signature creation, signature verification
- SHA-1, SHA-256, and MD5 hashing
- Strong random number generation (if enabled)

Those services that use cryptographic keys[12] can use:

- keys already loaded in the TOE through the mechanism outlined in O.LOAD_KEYS, or;
- keys generated by the TOE itself, or;
- keys supplied by S.HOST_APPL.

The key generation services can:

- store generated keys in the TOE or;
- supply generated keys to S.HOST_APPL.

O.LOAD_APPLICATION
If enabled, anyone with logical access to the TOE can try to load Application Software D.DEP_APPL in the TOE. The TOE verifies whether this Application Software was signed by S.BKS_ADM.

- If this is the case, the Application Software is executed by transforming the passive data object D.DEP_APPL into the active subject S.DEP_APPL;
- If this is not the case, the Application Software is rejected.

O.LOAD_KEYS
S.CUST_ADM can load S.APPL_KEYS from a C-ZAM/DEP, protected from modification and/or disclosure.

O.BACKUP_KEYS
If enabled, anyone with logical access to the TOE can backup S.APPL_KEYS from the TOE protected from disclosure.
If enabled, anyone with these backed-up keys and logical access to the TOE can later restore these keys in the TOE, if this is enabled.

[12] Encryption/decryption, MAC generation/verification, signature creation/verification.

O.ERASE
If enabled, anyone with logical access to the TOE can:

- erase S.APPL_KEYS
- erase S.APPL_KEYS and S.DEP.APPL from the TOE.

without moving the TOE from the NONE state.

O.ADMIN
S.CUST_ADM can:

- Enable/disable/limit the loading of S.DEP_APPL;
- Enable/disable/limit the backup/restore of S.APPL_KEYS
- Enable/disable/limit the strong random number generation[13]

O.STRONG_RANDOM
Any keys and/or random numbers generated by the TOE shall meet the DIEHARD and [F140-2MPRL] tests for random numbers.

[13] In theory, any of the services provided in O.SERVICES can be enabled/disabled/limited, as the writer of the Application Software determines this. In this particular Application Software it was chosen to provide this only for random number generation.

4.2. SECURITY OBJECTIVES FOR THE ENVIRONMENT

OE.ADMIN
The TOE shall be administered in a secure manner. This includes:

- Only trustworthy personnel can fulfil the roles of INIT_ADM, BKS_ADM, and CUST_ADM. This personnel is adequately trained, and keeps confidential information, such as passwords, keys, PINs secret)
- A C-ZAM/DEP in INIT mode, a C-ZAM/DEP in BKS mode and a C-ZAM/DEP in CUST mode shall be available. As these serve as the means to authenticate INIT_ADM, BKS_ADM, and CUST_ADM these shall be kept secure by INIT_ADM, BKS_ADM, and CUST_ADM respectively.

OE.ENABLE_PROTECT
Whenever an operation is enabled, (see O.ADMIN) anyone with logical access to the TOE can perform that operation. The environment must therefore ensure that only authorised use is made of that operation. This means that S.CUST_ADM should only enable some[14] operations in a trusted environment.

OE.SIGN_SOFTWARE
S.BKS_ADM must ensure that the Application Software D.DEP_APPL:

- is correct
- has the right Capabilities LOADED/UNLOADED
- and suitably protects access all cryptographic keys before signing it.

OE.KEY_GEN
Any keys generated outside the TOE that are subsequently loaded in the TOE must be generated in a confidential way, be unique with a very high probability and cryptographically strong.

OE.RANDOM_TEST
The IT environment shall provide means to verify the correct operation of the random number generation of the TOE.

OE.TRAFFIC
The IT environment shall protect any sensitive data in transit (this depends on how S.HOST_APPL and S.DEP_APPL work together) between S.HOST_APPL and S.DEP_APPL.

[14] Which operations depends on the context in which the TOE is used.

OE.SECURE_ROOM

The TOE will be deployed in a server or workstation in a "server-room" environment that restricts physical access to only necessary personnel[15]. The physical security of the room will be similar to a typical banking/financial institution computer server room. The environmental conditions will be similar to a typical computer server room.

[15] Ideally restricted to only S.CUST_ADM, but in practice other personnel (other sysadmins, security guards, cleaning staff) will also enter.

5. IT SECURITY REQUIREMENTS

5.1. TOE SECURITY FUNCTIONAL REQUIREMENTS

To achieve the security objectives for the TOE, the TSF has to meet a number of functional requirements. These have been sorted into several groups to enhance readability and understandability of the requirements and enable easy comparison with the security objectives for the TOE. The groups are:

- *Services offered by the TOE*: SFRs for the various services offered by the TOE: cryptographical services, key generation services and random number generation services.
- *Loading Application Software*: SFRs that allow the TOE to load and erase software.
- *Loading, backing up, and erasing Application Keys*
- *Managing the TOE*: SFRs that allow administrators to manage the TOE and control access to its services.
- *Tampering and abnormal operating conditions*: SFRs protecting the TOE against physical tampering, unusual temperatures etc.
- *Architectural security* SFRs that ensure that the TOE cannot be corrupted or bypassed.

On the notation that is used for SFRs:

1. Whenever an iteration was used, the component is numbered FXX_XXX.1+1 to FXX_XXX+n (for the nth iteration). A similar numbering scheme was used for the elements in each component.
2. The refinement operation was used in many cases to make the requirements easier to read and understand. All these cases were indicated.

5.1.1. Services offered by the TOE

SFRs for AES services offered by the TOE	
FCS_CKM.1+1	AES key generation
FCS_COP.1+1	AES encryption and decryption
FCS_COP.1+2	AES CBC-MAC generation
SFRs for DES services offered by the TOE	
FCS_CKM.1+2	DES/3DES key generation
FCS_COP.1+3	DES/3DES encryption and decryption
FCS_COP.1+4	DES/3DES CBC-MAC generation
SFRs for RSA services offered by the TOE	
FCS_CKM.1+3	RSA key generation
FCS_COP.1+5	RSA encryption and decryption
FCS_COP.1+6	RSA signature generation and verification
SFRs for other services offered by the TOE	
FCS_COP.1+7	SHA-1, SHA-256, MD5 hash generation
FCS_RND.1	Random number generation

5.1.1.1. AES services

FCS_CKM.1+1 Cryptographic key generation (for AES)
FCS_CKM.1.1+1 The TSF shall generate cryptographic keys **by**[16] **random number generation** and specified cryptographic key sizes **128 bit, 192 bit and 256 bit**[17]**.**
Dependencies:
FCS_COP.1+1 and 1+2 Cryptographic operation
~~FCS_CKM.4 Cryptographic key destruction~~[18]
FMT_MSA.2 Secure security attributes (met in IT environment)

FCS_COP.1+1 Cryptographic operation (AES)
FCS_COP.1.1+1 The TSF shall perform **encryption/decryption** in accordance with **AES in ECB mode, CBC mode, or CFB mode** and cryptographic key **sizes 128 bit, 192 bit and 256 bit** that meet **[AES].**
Dependencies:
[FDP_ITC.1 Import of user data without security attributes **and**[19] FCS_CKM.1+1 Cryptographic key generation]
~~FCS_CKM.4 Cryptographic key destruction~~
FMT_MSA.2 Secure security attributes (met in IT environment)

FCS_COP.1+2 Cryptographic operation (AES CBC-MAC)
FCS_COP.1.1+2 The TSF shall perform **MAC generation/MAC verification** in accordance with **AES CBC-MAC** and cryptographic key sizes **128 bit, 192 bit and 256 bit** that meet **[CBC-MAC].**
Dependencies:
[FDP_ITC.1 Import of user data without security attributes **and** FCS_CKM.1+1 Cryptographic key generation]
~~FCS_CKM.4 Cryptographic key destruction~~
FMT_MSA.2 Secure security attributes (met in IT environment)

---

[16] "in accordance with a specified cryptographic key generation algorithm" was refined to "by" to make the requirements more readable (Editorial refinement). This refinement has been applied multiple times in this section.

[17] The assignment was [no standards] and this was subsequently refined away to make the requirements more readable. This refinement has been used for all iterations of FCS_CKM.1.

[18] This dependency has been removed, as the requirements FAU_ARP.1, FDP_RIP.1 and FDP_ACF.1+2 already cover the deletion of cryptographic keys. This was done multiple times.

[19] The TOE can use keys generated internally (FCS_CKM.1), or keys that are given to it explicitly (FDP_ITC.1), so both dependencies must be included. This double dependency has been used multiple times in this section.

5.1.1.2. DES services

FCS_CKM.1+2 Cryptographic key generation (for DES and 3DES)
FCS_CKM.1.1+2 The TSF shall generate cryptographic keys **by random key generation with checking for weak keys** and specified cryptographic key sizes **56 bit, 112 bit and 168 bit.**
Dependencies:
FCS_COP.1+3 and 1+4 Cryptographic operation
~~FCS_CKM.4 Cryptographic key destruction~~
FMT_MSA.2 Secure security attributes (met in IT environment)

FCS_COP.1+3 Cryptographic operation (DES/3DES)
FCS_COP.1.1+3 The TSF shall perform **encryption/decryption** in accordance with **DES/3DES in ECB mode, CBC mode or CFB mode** and cryptographic key sizes **56 bit (DES), 112 bit (Triple-DES with two keys) and 168 bit (Triple-DES with three keys)** that meet **[(3)DES].**
Dependencies:
[FDP_ITC.1 Import of user data without security attributes **and** FCS_CKM.1+2 Cryptographic key generation]
~~FCS_CKM.4 Cryptographic key destruction~~
FMT_MSA.2 Secure security attributes (met in IT environment)

FCS_COP.1+4 Cryptographic operation (DES/3DES CBC-MAC)
FCS_COP.1.1+4 The TSF shall perform **MAC generation/MAC verification** in accordance with **DES/3DES CBC-MAC** and cryptographic key sizes **56 bit (DES), 112 bit (Triple-DES with two keys) and 168 bit (Triple-DES with three keys)** that meet **[CBC-MAC].**
Dependencies:
[FDP_ITC.1 Import of user data without security attributes **and** FCS_CKM.1+2 Cryptographic key generation]
~~FCS_CKM.4 Cryptographic key destruction~~
FMT_MSA.2 Secure security attributes (met in IT environment)

5.1.1.3. RSA services

FCS_CKM.1+3 Cryptographic key generation (for RSA)
FCS_CKM.1.1+3 The TSF shall generate cryptographic keys **by random RSA key generation** and **all**[20] cryptographic key sizes **between 512 and 2048 bit.**
Dependencies:
FCS_COP.1+5 and 1+6 Cryptographic operation
~~FCS_CKM.4 Cryptographic key destruction~~
FMT_MSA.2 Secure security attributes (met in IT environment)

FCS_COP.1+5 Cryptographic operation (RSA)
FCS_COP.1.1+5 The TSF shall perform **encryption/decryption** in accordance with **RSA** and **all** cryptographic key sizes **between 512 bit and 2048 bit** that meet **[PKCS#1(5)].**
Dependencies:
[FDP_ITC.1 Import of user data without security attributes **and** FCS_CKM.1+3 Cryptographic key generation]
~~FCS_CKM.4 Cryptographic key destruction~~
FMT_MSA.2 Secure security attributes (met in IT environment)

FCS_COP.1+6 Cryptographic operation (RSA signatures)
FCS_COP.1.1+6 The TSF shall perform **signature generation/signature verification** in accordance with **RSA** and **all** cryptographic key sizes **between 512 bit and 2048 bit** that meet **[ISO/IEC 9796-1], [ISO/IEC 9796-2] and [PKCS#1(8)].**
Dependencies:
[FDP_ITC.1 Import of user data without security attributes **and** FCS_CKM.1+3 Cryptographic key generation]
~~FCS_CKM.4 Cryptographic key destruction~~
FMT_MSA.2 Secure security attributes (met in IT environment)

[20] Refined requirement to show that e.g. 513 bit keys can also be generated without giving the explicit list [512 bit, 513 bit, 514 bit, …, 2048 bit]. This refinement has been used multiple times in this section.

5.1.1.4. Other services

FCS_COP.1+7 Cryptographic operation (hashing)

FCS_COP.1.1+7 The TSF shall perform **hash generation** in accordance with **SHA-1, SHA-256 and MD5**[21] that meet **[SHA] (for SHA-1 and SHA-256) and [MD5].**

Dependencies:

~~[FDP_ITC.1 Import of user data without security attributes or FCS_CKM.1 Cryptographic key generation]~~

~~FCS_CKM.4 Cryptographic key destruction~~

~~FMT_MSA.2 Secure security attributes~~[22]

FCS_RND.1 Quality metrics for random numbers

FCS_RND.1.1 The TSF shall provide a mechanism for generating random numbers that meet **the statistical random number generator tests described in [F140-2MPRL] and the statistical tests described in [DIEHARD].**

FCS_RND.1.2 The TSF[23] shall be able to enforce the use of TSF-generated random numbers for **random number generation and key generation.**

Dependencies: FPT_TST.1 TSF testing (met in IT environment)

[21] Refined away the “cryptographic key sizes” since these algorithms do not use keys.

[22] All dependencies are unnecessary, since these algorithms do not use keys.

[23] AIS31 uses the phrase TSFs here instead of TSF. As TSF (TOE Security Functions) is already plural, this is an error, therefore the term TSF was used.

5.1.2. Loading Application Software

SFRs for loading an application	
FDP_ACC.1+1	Naming the policy for the loading of an application
FDP_ACF.1+1	Providing the rules for loading of an application
FDP_ITC.2	The actual loading of software
FDP_DAU.2	Ensuring that only software from a trusted developer can be loaded

FDP_ACC.1+1 Subset access control (Applications)
FDP_ACC.1.1+1 The TSF shall enforce the **DEP Application Policy** on **[S.CUST_ADM], [D.DEP_APPL, S.DEP_APPL, D.APPL_KEYS[24]] and [R.LOAD_APPL, R.ERASE_APPL]**.
Dependencies: FDP_ACF.1+1 Security attribute based access control

FDP_ACF.1+1 Security attribute based access control (Applications)
FDP_ACF.1.1+1 The TSF shall enforce the **DEP Application Policy** to objects based on **Capability**.
FDP_ACF.1.2+1 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

- **Anyone with logical access to the TOE can use R.LOAD_APPL on D.DEP_APPL if Capability(R.LOAD_APPL) is LOADED and S.DEP_APPL is not present[25]**
- **Anyone with logical access to the TOE can use R.ERASE_APPL on S.DEP_APPL and D.APPL_KEYS if Capability(R.ERASE_APPL) is LOADED[26]**

FDP_ACF.1.3+1, FDP_ACF.1.4+1 ---[27]
Dependencies:
FDP_ACC.1+1 Subset access control
FMT_MSA.3 Static attribute initialisation

[24] This is present because removing the Application Software will also remove all Application Keys

[25] One can only load Application Software if there is none yet loaded.

[26] One can only remove Application Software if it is loaded.

[27] There are no additional explicit rules, so these requirements were refined away (Editorial refinement).

FDP_ITC.2 Import of user data with security attributes

FDP_ITC.2.1 The TSF shall enforce the **DEP Application Policy** when importing **D.DEP_APPL**[28] from outside of the TSC.
FDP_ITC.2.2 The TSF shall use the **Capabilities**[29] associated with the imported **D.DEP_APPL.**
FDP_ITC.2.3 The TSF shall ensure that the protocol used provides for the unambiguous association between the **Capabilities** and the **D.DEP_APPL** received.
FDP_ITC.2.4 The TSF shall ensure that interpretation of the **Capabilities** of the imported **D.DEP_APPL** is as intended by the source of **D.DEP_APPL**.
FDP_ITC.2.5 The TSF shall enforce the following rules when importing **D.DEP_APPL** from outside the TSC:

- **D.DEP_APPL shall only be imported if D.DEP_APPL was signed by S.BKS_ADM**[30]
- **D.DEP_APPL shall be imported through the PCI bridge**

Dependencies:
[FDP_ACC.1+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~~][31]
~~FPT_TDC.1 Inter-TSF basic TSF data consistency~~[32]

FDP_DAU.2 Data authentication with identity of guarantor

FDP_DAU.2.1 The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of **D.DEP_APPL**.
FDP_DAU.2.2 The TSF **shall**[33] **verify** the validity of **D.DEP_APPL** and the identity of the user that generated the evidence.
Dependencies: FIA_UID.1 Timing of identification (met in IT environment)

---

[28] Refined user data controlled under the SFP, to show to which user data the requirement applies. This was done several times in this requirement.

[29] Refined security attribute to show what it applies to. This was done multiple times.

[30] See FDP_DAU.2 for reference.

[31] Only the integrity and authenticity of D.DEP_APPL is important and this is covered by FDP_DAU.2 (see the previous footnote). As the confidentiality is not important the dependency on trusted path/trusted channel is unnecessary.

[32] Applications are specifically written for the TOE and only for the TOE. There is therefore no special interpretation necessary for the security attributes and the dependency is therefore unnecessary.

[33] Refined requirement to make it more readable as the TSF itself is the only subject that verifies loaded software.

5.1.3. Loading, backing up, and deleting Application Keys

SFRs for loading, saving and deleting keys	
FDP_ACC.1+2	Naming the policy for the loading/backup/restore/erasing of application keys
FDP_ACF.1+2	Providing the rules for the loading/backup/restore/erasing of application keys
FDP_ITC.1	The actual loading of application keys
FTP_ITC.1	The trusted channel between TOE and C-ZAM/DEP allowing secure loading of application keys

FDP_ACC.1+2 Subset access control (Application Keys)
FDP_ACC.1.1+2 The TSF shall enforce the **DEP Application Key Policy** on **[S.CUST_ADM], [D.APPL_KEYS, D.BKUPD_KEYS] and [R_LOAD_KEYS, R_BACKUP_KEYS, R_RESTORE_KEYS, R_ERASE_KEYS]**.
Dependencies: FDP_ACF.1+2 Security attribute based access control

FDP_ACF.1+2 Security attribute based access control (Application Keys)
FDP_ACF.1.1+2 The TSF shall enforce the **DEP Application Key Policy** to objects based on **Capability**.
FDP_ACF.1.2+2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

- **S.CUST_ADM can use R.LOAD_KEYS on D.APPL_KEYS if Capability(R.LOAD_KEYS) is LOADED and S.DEP_APPL is present**[34]
- **Anyone with logical access to the TOE can create D.BKUPD_KEYS by using R.BACKUP_KEYS on D.APPL_KEYS if Capability(R.BACKUP_KEYS) is LOADED and S.DEP_APPL is present**
- **Anyone with logical access to the TOE can modify D.APPL_KEYS by using R.RESTORE_KEYS on D.BKDUP_KEYS if Capability(R.RESTORE_KEYS) is LOADED and S.DEP_APPL is present**
- **Anyone with logical access to the TOE can use R.ERASE_KEYS on D.APPL_KEYS if Capability(R.ERASE_KEYS) is LOADED**

FDP_ACF.1.3 ---[35]
FDP_ACF.1.4 The TSF shall explicitly deny access of subjects and objects based on

- **Nobody is allowed to read D.BKDUP_KEYS except the TSF.**

Dependencies:
FDP_ACC.1+2 Subset access control
FMT_MSA.3 Static attribute initialisation

[34] I.e. one can only load/backup/restore keys if software is loaded.

[35] There are no additional explicit rules, so these requirements were refined away (Editorial refinement).

FDP_ITC.1 Import of user data without security attributes

FDP_ITC.1.1 The TSF shall enforce the **DEP Application Key Policy** when importing **D.APPL_KEYS**[36] from outside of the TSC.

FDP_ITC.1.2 The TSF shall ignore any security attributes associated with **D.APPL_KEYS** when imported from outside the TSC.

FDP_ITC.1.3 The TSF shall[37] **only import D.APPL_KEYS** from outside the TSC from the **C-ZAM/DEP through the trusted channel**.

Dependencies:

[FDP_ACC.1+2 Subset access control ~~or FDP_IFC.1 Subset information flow control~~]
~~FMT_MSA.3 Static attribute initialisation~~[38]

FTP_ITC.1 Inter-TSF trusted channel

FTP_ITC.1.1 The TSF shall provide a communication channel between itself and a **C-ZAM/DEP**[39] 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 **the C-ZAM/DEP** to initiate communication via the trusted channel **to**[40]**:**

- **import D.APPL_KEYS into the TSF;**
- **MoveToInit, MoveToCust management functions;**[41]
- **modify Capabilities.**[42]

FTP_ITC.1.3 The TSF shall **not**[43] initiate communication via the trusted channel.

---

[36] Refined user data controlled under the SFP, to show to which user data the requirement applies. This was done several times in this requirement.

[37] Requirement was reworded to make it easier to read.

[38] As keys have no security attributes, this dependency is not useful.

[39] Refined trusted IT-product to show to which product the requirement applies. This refinement has been applied several times in this section.

[40] Refined the requirement to show what the trusted channel is used for.

[41] See FMT_SMF.1 and FMT_MOF.1

[42] See FMT_MSA.1, FMT_SAE.1, FMT_SAE.2

[43] Refined the requirement to make it easier to read.

5.1.4. Management of the TOE

SFRs for management of the TOE	
FMT_SMR.1	The different administrative roles that can manage the TOE
FIA_UID.2	These roles must be identified before they can do anything
FIA_UAU.2	These roles must be authenticated before they can do anything
FMT_MSA.1	How S.CUST_ADM can enable/disable operations
FMT_SAE.1	How S.CUST_ADM can limit operations in time
FMT_SAE.2 (extended SFR)	How S.CUST_ADM can limit the number of times an operation is used
FDP_ACC.1+3	Naming the policy that controls access to operations
FDP_ACF.1+3	The rules that allow access to operations
FMT_MSA.3	The default access to operations
FMT_SMF.1	The different management operations of the TOE
FMT_MOF.1	Which role is allowed to use certain management operations

FMT_SMR.1 Security roles
FMT_SMR.1.1 The TSF shall maintain the roles **S.INIT_ADM, S.BKS_ADM, and S.CUST_ADM**.
FMT_SMR.1.2 The TSF shall be able to associate users with roles.
Dependencies: FIA_UID.1 Timing of identification (met by FIA_UID.2)

FIA_UID.2 User identification before any action
FIA_UID.2.1 The TSF shall require **S.INIT_ADM, S.BKS_ADM and S.CUST_ADM**[44] to identify **themselves**[45] before allowing any other TSF-mediated actions on behalf of **those users**.

FIA_UAU.2 User authentication before any action
FIA_UAU.2.1 The TSF shall require **S.INIT_ADM, S.BKS_ADM and S.CUST_ADM** to be successfully authenticated before allowing any other TSF-mediated actions on behalf of **those users**.
Dependencies: FIA_UID.1 Timing of identification (met by FIA_UID.2)

[44] The requirement was refined to show to which users the requirement applies. A similar refinement was made for FIA_UAU.2

[45] The sentence was editorially refined to correct the grammar. A similar refinement was made in FIA_UAU.2

FMT_MSA.1 Management of security attributes

FMT_MSA.1.1 The TSF shall enforce the **DEP Application Policy, the DEP Application Key Policy and the DEP Executing Policy** to restrict the ability to **modify** the security attributes **Capability** to **S.CUST_ADM**.

Dependencies:

[FDP_ACC.1+1-3 Subset access control ~~or FDP_IFC.1 Subset information flow control~~]

FMT_SMF.1 Specification of management functions

FMT_SMR.1 Security roles

FMT_SAE.1 Time-limited authorisation

FMT_SAE.1.1 The TSF shall restrict the capability to specify an expiration time for **Capability** to **S.CUST_ADM.**

FMT_SAE.1.2 For each of these security attributes, the TSF shall be able to **set the Capability to UNLOADED** after the expiration time for the indicated security attribute has passed.

Dependencies:

FMT_SMR.1 Security roles

FPT_STM.1 Reliable time stamps

FMT_SAE.2 Use-limited authorisation

FMT_SAE.2.1 The TSF shall restrict the capability to specify a maximum number of uses for **Capability** to **S.CUST_ADM.**

FMT_SAE.2. For each of these security attributes, the TSF shall be able to **set Capability to UNLOADED** after the **operation** to which the indicated security attribute belongs has been used the maximum number of times.

Dependencies:

FMT_SMR.1 Security roles

FDP_ACC.1+3 Subset access control

FDP_ACC.1.1+3 The TSF shall enforce the **DEP Executing Policy** on **[S.HOST_APPL], [] and [R.RND].**

Dependencies: FDP_ACF.1+3 Security attribute based access control

FDP_ACF.1+3 Security attribute based access control

FDP_ACF.1.1+3 The TSF shall enforce the **DEP Executing Policy** to objects based on **Capability**.

FDP_ACF.1.2+3 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

- **Anyone with logical access to the TOE can perform R.RND[46] if the Capability for R.RND is LOADED**

FDP_ACF.1.3+3, FDP_ACF.1.4+3 ---[47]

Dependencies:

FDP_ACC.1+3 Subset access control

FMT_MSA.3 Static attribute initialisation

FMT_MSA.3 Static attribute initialisation

FMT_MSA.3.1 The TSF shall enforce the **DEP Application Policy, the DEP Application Key Policy and the DEP Executing Policy** to **set**[48]

- **Capability for R.LOAD_APPL, R.ERASE_APPL, R.LOAD_KEYS, R.ERASE_KEYS, R.BACKUP_KEYS, R.RESTORE_KEYS and R.RND to UNLOADED**

FMT_MSA.3.2 The TSF shall allow[49] **nobody** to specify alternative initial values to override the default values when an object or information is created.

Dependencies:

FMT_MSA.1 Management of security attributes

FMT_SMR.1 Security roles

---

[46] In theory, any of the services in R.SERV_EXT can have a Capability, as the writer of the Application Software determines this. In this Application Software (see the subsection Library EVAL in section 2.3.1 for more information) it was chosen to provide only R.RND with a Capability security attribute.

[47] There are no additional explicit rules, so these elements were refined away.

[48] This requirement was refined to replace the confusing terms permissive/restrictive with the actual values of the security attribute. In addition the term “security attributes that are used to enforce the SFP” was refined to the actual security attribute.

[49] “the” removed to make the sentence grammatically correct.

FMT_SMF.1 Specification of Management Functions[50]

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

- **MoveToNONE** bringing the TOE back to Authority Level NONE, this will clear the TOE of S.DEP_APPL and D.APPL_KEYS
- **AssignCustomer** assigning the TOE to a specific customer
- **SetMode** set the TOE to LIV, TST or DEV mode
- **MoveToINIT** bringing the TOE to Authority Level INIT
- **MoveToBKS** bringing the TOE to Authority Level BKS
- **MoveToCUST** bringing the TOE to Authority Level CUST

FMT_MOF.1 Management of security functions behaviour

FMT_MOF.1.1 The TSF shall restrict the ability to **enable and disable** the functions:

- **MoveToNONE** **to** **all**[51]
- **AssignCustomer** **to** **all**
- **SetMode** **to** **all**
- **MoveToINIT** **to** **all**
- **MoveToBKS** **to** **S.INIT_ADM**
- **MoveToCUST** **to** **S.BKS_ADM**

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

[50] This is **not** an extended requirement, but was taken from Interpretation 65. which made this requirement part of the CC.

[51] Anyone can move the TOE back to NONE by triggering an alarm (hitting the TOE). The restrictions to “all” are there to provide a complete overview.

5.1.5. Tampering and abnormal operating conditions

SFRs for tampering and abnormal operating conditions	
FPT_PHP.3	Resistance to physical attacks (penetrating or removing the cover)
FAU_GEN.1	Logging tamper events (physical attacks and abnormal operating conditions)
FPT_STM.1	Generating an accurate timestamp for logging purposes
FAU_SAA.1	Detecting when a tamper event has taken place
FAU_ARP.1	Reacting on a detected tamper event (clearing the DEP)
FDP_RIP.1	Ensuring that when the DEP is cleared, the data cannot be retrieved

FPT_PHP.3 Resistance to physical attack

FPT_PHP.3.1 The TSF shall resist **physical penetration, chemical penetration and removal of**[52] **the cover** by **generating a tamper event**[53][54]**.**

[52] Editorial refinement to make the requirement more readable.

[53] Refinement to clarify how it will be ensured that the TSP is not violated.

[54] See FAU_GEN.1 for a list of all tamper events.

FAU_GEN.1 Audit data generation

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions;

b) ---[55]

c) **tamper events (physical penetration of the cover, chemical penetration of the cover, removal of the cover, unusual temperatures, unusual voltages, removal of TOE from its PCI slot, unusual physical acceleration of the TOE)**

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information:

a) Date and time of the event, type of event[56] [57]

b) ---[58]

Dependencies: FPT_STM.1 Reliable time stamps

FPT_STM.1 Reliable time stamps

FPT_STM.1.1 The TSF shall be able to provide reliable time stamps for its own use.

FAU_SAA.1 Potential violation analysis

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

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

a) **Occurrence[59]** of **tamper events[60]**;

b) **---**[61]

Dependencies: FAU_GEN.1 Audit data generation

---

[55] Element b) was removed (CC Part 2 paragraph 567)

[56] Subject identity is not recorded: audit functions are always started/stopped by S.CUST_ADM, and the identity of subjects causing tamper events is always unknown.

[57] Success or failure is not recorded: it is not useful to record unsuccessful starting or stopping of audit functions, and “success or failure” is undefined w.r.t. a tamper event.

[58] Element b) was removed, as no level was selected no information on the functional components was recorded.

[59] Refined “accumulation or combination” to “occurrence” as the TOE only needs a single tamper event to trigger.

[60] Refined “known to indicate a potential security violation” as all tamper events indicate a potential security violation.

[61] No other rules exist.

FAU_ARP.1 Security alarms

FAU_ARP.1.1 The TSF shall[62]:

1. **deallocate S.DEP_APPL and D.APPL_KEYS from the TOE resources used to store them in**
2. **set the Capability of R.LOAD_APPL, R.ERASE_APPL, R.LOAD_KEYS, R.BACKUP_KEYS, R.RESTORE_KEYS, R.ERASE_KEYS to UNLOADED**

upon detection of a potential security violation.

Dependencies: FAU_SAA.1 Potential violation analysis

FDP_RIP.1 Subset residual information protection

FDP_RIP.1.1 **When a potential security violation is detected,[63]** the TSF shall ensure that any previous information content of a resource is made unavailable upon the **deallocation of the resource** from[64] **S.DEP_APPL and D.APPL_KEYS.**

[62] Removed “take” to make the sentence grammatically correct.

[63] Added a refinement to show that FDP_RIP only activates when a tamper event occurs, and not during normal operation of the TOE (e.g. when using R.ERASE_KEYS)

[64] Removed “the following objects” to make the requirement easier to read.

5.1.6. Architectural security

SFRs for Architectural Security	
FPT_SEP.1	Resistance to logical attacks
FPT_RVM.1	Ensuring that the TSF cannot be bypassed

FPT_SEP.1 TSF domain separation

FPT_SEP.1.1 The TSF shall maintain a security domain for its own execution that protects it from interference and tampering by untrusted subjects.

FPT_SEP.1.2 The TSF shall enforce separation between the security domains of subjects in the TSC.

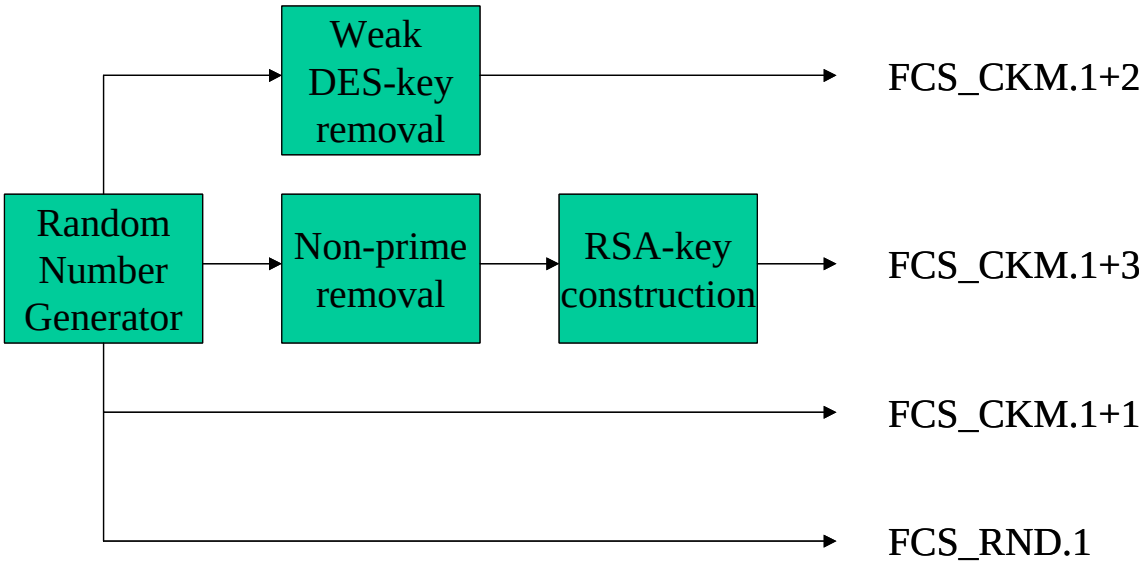
FPT_RVM.1 Non-bypassability of the TSP

FPT_RVM.1.1 The TSF shall ensure that TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed.

5.1.7. Strength-of-function claim

The minimum strength-of-function level for the DEP/PCI is: not applicable.

The TOE does have SFRs that rely on probabilistic/permutational mechanisms, as the following figure shows:



From the figure it can be seen that all SFRs depend on a single probabilistic/ permutational random generator mechanism. However, no strength-of-function claim is made for this random generator mechanism, and therefore no minimum strength-of-function level applies.

5.2. TOE SECURITY ASSURANCE REQUIREMENTS

The TOE security assurance requirements are conformant to the CC Evaluation Assurance Level EAL3 + ADV_FSP.2. In detail the following Security Assurance Requirements are chosen for the TOE:

Components for Configuration management (**Class ACM**)
- ACM_CAP.3 Authorisation controls
- ACM_SCP.1 TOE CM coverage

Components for Delivery and operation (**Class ADO**)
- ADO_DEL.1 Delivery procedures
- ADO_IGS.1 Installation, generation, and start-up procedures

Components for Development (**Class ADV**)
- ***ADV_FSP.2 Fully defined external interfaces***[65]
- ADV_HLD.2 Security enforcing high-level design
- ADV_RCR.1 Informal correspondence demonstration

Components for Guidance documents (**Class AGD**)
- AGD_ADM.1 Administrator Guidance
- AGD_USR.1 User guidance

Components for Life cycle support (**Class ALC**)
- ALC_DVS.1 Identification of security measures

Components for Tests (**Class ATE**)
- ATE_COV.2 Analysis of coverage
- ATE_DPT.1 Testing: high-level design
- ***ATE_FUN.1 Functional testing***[66]
- ATE_IND.2 Independent testing – sample

Components for Vulnerability assessment (**Class AVA**)
- AVA_MSU.1 Examination of guidance
- AVA_SOF.1 Strength of TOE security function evaluation
- AVA_VLA.1 Developer vulnerability analysis

ATE_FUN.1 is refined as follows:
ATE_FUN.1.3C The test procedure descriptions shall identify the tests to be performed and describe the scenarios for testing each security function. These scenarios shall include any ordering dependencies on the results of other tests**. The scenarios for RND_1**[67] **shall include the entire DIEHARD test suite and all tests in [F140-2MPRL].**

[65] This is an augmentation of EAL3.

[66] This requirement is refined.

[67] See section 6.1.1 for a definition of this function.

5.3. SECURITY REQUIREMENTS FOR THE IT ENVIRONMENT

To achieve the IT security objectives for the environment, the IT environment of the TOE has to fulfil the following functional requirements.

FMT_MSA.2 Secure security attributes

FMT_MSA.2.1 The **D.DEP_APPL Author**[68] shall ensure that only secure values are accepted for security attributes **of D.APPL_KEYS**[69].

Dependencies:

~~ADV_SPM.1 Informal TOE security policy model~~
~~[FDP_ACC.1 Subset access control or~~
~~FDP_IFC.1 Subset information flow control]~~
~~FMT_MSA.1 Management of security attributes~~
~~FMT_SMR.1 Security roles~~[70]

Note: Access to cryptographic keys is arranged completely by the D.DEP_APPL author. This author is responsible that keys are used in the correct way. As far as the TOE is concerned these keys therefore have no security attributes.

[68] This refinement defines the parts of the IT environment that must meet this requirement.

[69] This refinement defines the security attributes that this requirement applies to.

[70] The model used to define what values are secure, the policy that these secure values help implement, how these values are managed during the writing of the Application Software and the various roles involved in writing the Application Software are all the responsibility of the author of the Application Software and hence fall outside the scope of this ST. All these dependencies were therefore deleted.

FPT_TST.1 TSF testing

FPT_TST.1.1 The **IT environment** shall run a suite of ~~self~~[71] tests **at the request of the authorised user**[72] to demonstrate the correct operation of the **random number generator part of the TSF**[73]**.**

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

~~FPT_TST.1.3 The TSF shall provide authorised users with the capability to verify the integrity of stored TSF executable code.~~[74]

Dependencies: ~~FPT_AMT.1 Abstract machine testing~~[75]

Note: This requirement will be implemented as follows: An authorised user uses the TOE to generate a large amount of random data. He can then run a test program consisting of the DIEHARD ~~and FIPS 140-2 Level 4~~ tests for random numbers to test whether this data is truly random. This test program is delivered with the TOE.

[71] As the environment performs the tests, the word “self” was removed.

[72] How this user is authorised (i.e. provided with the test program and logical TOE access) falls outside the scope of this ST.

[73] This refinement defines the parts of the TSF that must meet this requirement.

[74] These two elements do not make sense when applied to the IT environment.

[75] The concept “abstract machine” is not relevant in the context of the IT environment, so the dependency is not useful.

5.4. EXPLICITLY STATED REQUIREMENTS

In this ST two SFRs have been used that are not in Part 2:

- FMT_SAE.2 Use-limited authorization;
- FCS_RND.1 Quality metrics for random numbers.

5.4.1. FMT_SAE.2 Use-limited authorization

FMT_SAE.2 is defined as part of the FMT_SAE family, because it is closely linked with FMT_SAE.1 Time-limited authorization, which allows specification of a security attribute that expires after a given amount of time.

FMT_SAE.2 Use-limited authorization is intended for operations, subjects and/or objects that can only be used a limited amount of times, i.e. after they have been used x times, they can no longer be used.

No other requirement in Part 2 could be found to clearly express this.

FMT_SAE.2 Use-limited authorization

Hierarchical to: No other components.

FMT_SAE.2.1 The TSF shall restrict the capability to specify a maximum number of uses for [assignment: *list of security attributes for which maximum is to be supported*] to [assignment*: the authorized identified roles*]

FMT_SAE.2.2 For each of these security attributes, the TSF shall be able to [assignment: *list of actions to be taken*] after the [selection: operation, subject, object] to which the indicated security attribute belongs has been used the maximum number of times.

Dependencies:

FMT_SMR.1 Security roles

The assurance requirements are applicable and appropriate to support this SFR because it closely resembles FMT_SAE.1 and therefore requires no special assurance techniques (otherwise FMT_SAE.1 would have an assurance dependency).

The dependency on FMT_SMR.1 is identical to that in FMT_SAE.1. The other dependency of FMT_SAE.1 (FPT_STM.1) is not a dependency of FMT_SAE.2, as FMT_SAE.2 does not use time in any way.

5.4.2. FCS_RND.1 Quality metrics for random numbers[76].

FCS_RND.1 is defined in order to avoid being restricted to the FIA class when using random numbers and to explicitly describe the usage of random numbers for key generation (FCS_CKM.1) or in cryptographic algorithms or protocols (FCS_COP.1). In addition, FCS_RND.1 provides the connection to start-up tests and online tests for the random number generator.

No other requirement in Part 2 could be found to clearly express this. No family could be found to fit this requirement in. As it is strongly related to cryptography, it was placed as a new family in the FCS (Cryptographic Support) class.

FCS_RND.1 Quality metrics for random numbers
Hierarchical to: No other components.
FCS_RND.1.1 The TSF shall provide a mechanism for generating random numbers that meet [assignment: *a defined quality metric*].
FCS_RND.1.2 The TSF shall be able to enforce the use of TSF-generated random numbers for [assignment: *list of TSF* functions]
Dependencies:
FPT_TST.1 TSF testing

The assurance requirements are applicable and appropriate to support this SFR because it closely resembles FIA_SOS.2 and therefore requires no special assurance techniques (otherwise FIA_SOS.2 would have an assurance dependency). It has FPT_TST.1 as dependency because random number generators can easily obtain a bias without this being visible to users.

[76] This section was copied from AIS31. A slight modification was made ("TSFs" was changed to "TSF" to adhere to proper CC grammar.

6. TOE SUMMARY SPECIFICATION

6.1. TOE SECURITY FUNCTIONS

6.1.1. Cryptographic service functions

The cryptographic service functions are only available in CUST Authority Level. All cryptographic service functions are provided to S.HOST_APPL. All cryptographic service functions using keys can be either performed with keys stored in the TOE, or with keys provided by S.HOST_APPL.

Name	Function
KEY_1	The TOE provides AES cryptographic key generation services
CRYPT_1	The TOE provides AES encryption/decryption services
CRYPT_2	The TOE provides AES CBC-MAC generation services
KEY_2	The TOE provides DES/3DES cryptographic key generation services
CRYPT_3	The TOE provides DES/3DES encryption/decryption services
CRYPT_4	The TOE provides DES/3DES CBC-MAC generation services
KEY_3	The TOE provides RSA cryptographic key generation services
CRYPT_5	The TOE provides RSA encryption/decryption services
CRYPT_6	The TOE provides RSA signature creation/verification services
CRYPT_7	The TOE provides SHA-1, SHA-256 and MD5 hash generation services
RND_1	The TOE provides random number generation services

6.1.2. Loading and saving security functions

These functions are only available in CUST Authority Level.

Name	Function
LOAD_1	(if no Application Software has been loaded) The TOE can load Application Software through the PCI-bridge. The TOE rejects Application Software not signed by BKS_ADM.
LOAD_2	(if Application Software is loaded) S.CUST_ADM can load Application Keys in the TOE. This function can only be performed through the trusted channel provided by CZAM_1.
BACKUP_1	The TOE can backup Application Keys to the PCI-bridge. Only the TOE can read these Application Keys.
BACKUP_2	The TOE can restore previously backed-up Application Keys from the PCI-bridge.
ERASE_1	(if Application Software has been loaded) Application Software and all Application Keys can be deleted together from the TOE without moving the TOE to NONE Authority Level.
ERASE_2	Application Keys can be deleted from the TOE without deleting the Application Software and without moving the TOE to NONE Authority Level.

6.1.3. Managing the TOE

The Authority levels where these functions are available are given with each function.

Name	Function
CZAM_1	(in INIT, BKS and CUST Authority Levels) A C-ZAM/DEP can set-up a trusted channel between that C-ZAM/DEP and the TOE. Through this trusted channel the TOE can: • identify and authenticate S.INIT_ADM (in Authority Level INIT) • identify and authenticate S.BKS_ADM (in Authority Level BKS) • identify and authenticate S.CUST_ADM (in Authority Level CUST)
MODE_1	(all Authority Levels) When the TOE is brought to NONE level the Application Software and all Application Keys are irretrievably deleted.
MODE_2	(only in Authority Level NONE) The TOE can be set to LIV mode and be assigned to a specific customer. This mode and customer cannot be changed, except by resetting the TOE to NONE Authority Level.
MODE_3	(only in Authority Level NONE, and when a customer and mode have been assigned). The TOE can be brought to Authority Level INIT.
MODE_4	(only in Authority Level INIT) S.INIT_ADM can bring the TOE to Authority Level BKS. This function can only be performed through the trusted channel provided by CZAM_1.
MODE_5	(only in Authority Level BKS) S.BKS_ADM can bring the TOE to Authority Level CUST. This function can only be performed through the trusted channel provided by CZAM_1.
MANAG_1	(only in Authority Level CUST) S.CUST_ADM is able to enable/disable/limit individual RND_*[77], LOAD_*, BACKUP_*, and ERASE_* functions. Limiting functions means either enabling a function with a time-limit or with a certain maximum number of uses. This function can only be performed through the trusted channel provided by CZAM_1.
MANAG_2	RND_*, LOAD_*, BACKUP_* and ERASE_* security functions are disabled by default.

[77] The application writer determines which external services can be disabled or enabled. In the Eval Library only the Random number generation function can be enabled/disabled.

6.1.4. Tampering/Abnormal conditions security functions

These functions are available in all Authority Levels.

Name	Function
ALARM_1	The TOE detects tamper events: physical penetration, chemical penetration, removal of the cover, unusual temperatures, unusual voltage, removal of the TOE from its PCI slot, unusual physical acceleration of the TOE.
LOG_1	The TOE logs tamper events resulting from ALARM_1, with type and time.
REACT_1	The TOE moves back to NONE Authority Level upon detecting a tamper event.
PROT_1	The TOE is enclosed in a hard casing. This casing is difficult to physically remove, bypass or penetrate (tamper resistant).

6.1.5. Architectural security

These functions are available in all Authority Levels.

Name	Function
ARCH_1	The TOE's own internal code and data cannot be disclosed or modified.
ARCH_2	The TOE protects the Application Software and Application Keys against modification (except through the LOAD_*, ERASE_* and BACKUP_* functions).
ARCH_3	The TOE protects the Application Keys against disclosure.
ARCH_4	The TOE ensures that its security functions cannot be bypassed.

6.1.6. Probabilistic functions and mechanisms

These are not relevant. See section 5.1.7 for details.

6.2. ASSURANCE MEASURES

Appropriate assurance measures are employed to satisfy the security assurance requirements. The following list gives a mapping between the assurance requirements and the documents containing the information needed for the fulfilment of the respective requirement.

Configuration Management (ACM) assurance measures
The documents containing the description of the configuration management system and how it is used are:

- DEP Group – Configuration Management, version 1.0 (4)
- DEP/PCI – Manufacturing Security Guidelines, version 1.0 (5)
- DEP Group – VSS Configuration, version 1.0 (9)
- Visual SourceSafe Helpfile, version 6.0
- DEP/PCI – Configuration CommonC, version 3.0 (12)
- DEP/PCI – Test Overview, version 1.0 (4)

Delivery and Operation (ADO) assurance measures
The documents containing the description of all steps necessary for secure installation, generation and start-up of the TOE are:

- DEP/PCI – Manufacturing Security Guidelines, version 1.0 (5)
- DEP/PCI – Delivery Procedures, version 1.0 (5)
- labels DCC5.xls
- DEP/NT Documentation – DEP/NT Banksys Security Officers Guide, version 02.01
- DEP/NT Documentation – DEP/NT Management Tool User Manual, version 01.03
- DEP/NT Documentation – DEP/NT C-ZAM/DEP User Manual, version 02.03
- DEP/NT Documentation – DEP/NT Installation Guide, version 02.01

Development (ADV) assurance measures
The development documentation can be found in:

- DEP/PCI – Functional Specifications Overview, version 1.0 (4)
- DEP/PCI – High Level Design Overview, version 1.0 (6)
- DEP/PCI – Functional Specifications Hardware Security, version 1.0 (8)
- DEP/PCI – High Level Design Hardware Security, version 1.2 (2)
- DEP/PCI – Development Documentation – Erratum, version 1.0 (1)
- DEP/PCI – Corr Table: TOE Summary Specifications – Functional Specifications, version 1.0 (4)
- DEP/PCI – Corr Table: TOE Functional Specifications – HLD, version 1.0 (5)
- DEP/PCI – Vulnerability Analysis, version 1.0 (6)
- DEP/NT Documentation – DEP/NT DS/3 Principles, version 02.02
- CZD – DEP Interface Library for DEP – Reference DFS Manual, version 3.0 (13)
- Subset STD Library for DEP – Ref DFS Manual, version 3.5 (8)
- DEP-PC Project – DS3_ALARM sub-project DFS, version 1.4

- DEP-PC Project – DS3_ALARM sub-project ADD, version 1.4
- PCI Interface Board Specification, Issue 3.2
- DEP/PCI – Kiss Communication Protocol, version 1.3 (2)
- DEP Cryptographic ToolBox, version 1.0 (2)
- Common Criteria Software – Integration Manual, version 1.0 (10)
- Subset of Eval Library for DEP – Reference DFS Manual, version 2.0 (12)

Guidance (AGD) assurance measures
The documents containing the guidance for the banksys and customers administrators are:
- DEP/PCI – Customer Security Officer: Guidelines, version 1.0 (10)
- DEP/PCI – Customer Host Programmers Guidelines, version 1.0 (1)
- DEP/NT Documentation – DEP/NT C-ZAM/DEP User Manual, version 02.03
- DEP/NT Documentation – DEP/NT PC-AUX Program User Manual, version 02.01
- DEP/NT Documentation – DEP/NT Host Interface Supervision User Manual, version 02.01
- DEP/NT Documentation – DEP/NT DEP Handler Supervision User Manual, version 02.03
- Subset STD Library for DEP – Ref DFS Manual, version 3.5 (8)
- Common Criteria Software – Integration Manual, version 1.0 (10)
- Subset of Eval Library for DEP – Reference DFS Manual, version 2.0 (12)
- DEP/PCI – Random Test, version 1.0 (1)
- DEP/PCI – Guidance Documentation – Erratum, version 1.0 (1)

For information purposes, the following additional documents are available:
- DEP/NT Documentation – Introduction to DEP/NT, version 02.01
- DEP/NT Documentation – DEP/NT Glossary, version 02.03
- DEP/NT Documentation – DEP/NT DS/3 Principles, version 02.02

Life Cycle (ALC) assurance measures
The physical, procedural, personnel and other security measures applied by banksys can be found in:
- banksys – Physical Security Measures, version 1.0 (3)
- DEP Group – Security Measures, version 1.0 (2)
- DEP Group – Configuration Management, version 1.0 (4)
- DEP Group – VSS Configuration, version 1.0 (9)
- DEP/PCI – Manufacturing Security Guidelines, version 1.0 (5)

Test (ATE) assurance measures
The test documents and results are:
- DEP/PCI – Test Overview, version 1.0 (4)
- DEP/PCI – Random Test, version 1.0 (1)
- DepPciHW_V1.0.3..zip: hardware test data (test correspondence table, test scripts, test reports)

- DepPciAla_V2.0.1.zip: Alarm interface test data (test correspondence table, test scripts, test scenarios, test reports)
- DepPciAux1_V1.0.4.zip: AUX1 test data (test correspondence table, test scripts, test scenarios, test reports)
- DepPciBoot_V1.0.4.zip: Boot Software test data (test correspondence table, test scripts, test scenarios, test reports)
- DepPciCzd_V1.0.4.zip: C-ZAM/DEP interface test data (test correspondence table, test scripts, test scenarios, test reports)
- DepPciEval_V1.0.5.zip: Eval Library test data (test correspondence table, test scripts, test scenarios, test reports)
- DepPciStd_V1.0.5.zip: Std Library test data (test correspondence table, test scripts, test scenarios, test reports)
- Libs_Banksys_V1.0.4.zip: Common Test Libraries
- DEP/PCI – Test Documentation – Erratum, version 1.0 (1)
- User Manual – TheTestFactory ScriptWriter & ScenarioPlayer, version 1.1 (6)
- User Manual – TheTestFactory Test Script Language, version 1.1 (5)

Vulnerability Assessment (AVA) assurance measures

The vulnerability analysis documents are:

- DEP/PCI – Customer Security Officer: Guidelines, version 1.0 (10)
- DEP/PCI – Vulnerability Analysis, version 1.0 (6)
- DEP Group – Development Resources, version 1.0 (1)

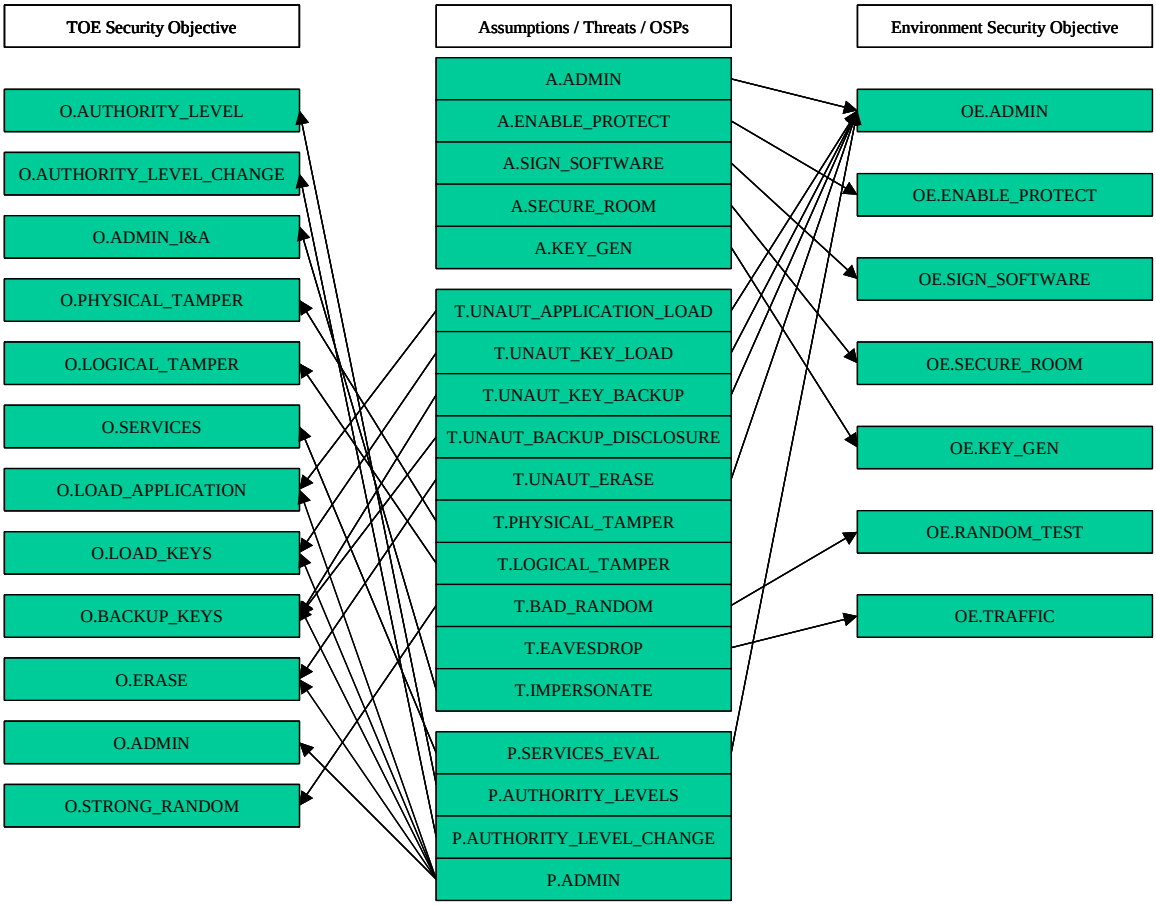
7. PP CLAIMS

This Security Target TOE does not claim conformance to any Protection Profile.

8. RATIONALE

8.1. SECURITY OBJECTIVES RATIONALE

For each assumption, threat and OSP we demonstrate that it is met by the security objectives. The tracings are provided in the following figure.



The individual rationales demonstrating that the objectives are as follows:

A.ADMIN
This assumption is being met by OE.ADMIN, which is a direct translation.

A.ENABLE_PROTECT
This assumption is being met by OE.ENABLE_PROTECT, which is a direct translation.

A.SIGN_SOFTWARE
This assumption is being met by OE.SIGN_SOFTWARE, which is a direct translation.

A.KEY_GEN
This assumption is being met by OE.KEY_GEN, which is a direct translation.

A.SECURE_ROOM
This assumption is being met by OE.SECURE_ROOM, which is a direct translation.

T.UNAUT_APPLICATION_LOAD
This threat is countered by O.LOAD_APPLICATION, showing that applications must be signed, thus preventing "illegal" applications and O.ADMIN, showing that the loading of applications can be disabled by S.CUST_ADM, thus preventing the loading of "legal" but old/different applications.

T.UNAUT_KEY_LOAD
This threat is countered by O.LOAD_KEYS showing that an attacker would need the C-ZAM/DEP of S.CUST_ADM, and OE.ADMIN, showing that S.CUST_ADM will keep this device secure. Additionally, O.ADMIN shows that the loading of keys can be disabled by S.CUST_ADM.

T.UNAUT_KEY_BACKUP
This threat is countered by O.BACKUP_KEYS and O.ADMIN showing that the backing up and restoring of keys can be turned off.

T.UNAUT_BACKUP_DISCLOSURE
This threat is countered by O.BACKUP_KEYS, showing that the backed up keys are protected against disclosure.

T.UNAUT_ERASE
This threat is countered by O.ERASE and O.ADMIN showing that the erasing of keys and/or software can be turned off.

T.PHYSICAL_TAMPER
This threat is countered by O.PHYSICAL_TAMPER showing that the TOE is protected against physical tampering attacks.

T.LOGICAL_TAMPER
This threat is countered by O.LOGICAL_TAMPER showing that the TOE is protected against logical tampering attacks.

T.BAD_RANDOM
This threat is countered by:

- O.STRONG_RANDOM, showing that random number generation and key generation are both strong enough to meet two well-known high-quality public-domain test suites. This covers the quality of random numbers at the time the TOE is evaluated.
- OE.RANDOM_TEST allowing the testing of the quality of the random number generator at any time during the lifetime of the TOE.

T.EAVESDROP
This threat is countered by OE.TRAFFIC, showing that if the traffic between the TOE and S.HOST_APPL is confidential, there should be adequate measures in the environment.

T.IMPERSONATE
This threat is countered by O.ADMIN_I&A, showing how S S.INIT_ADM, S.BKS_ADM or S.CUST_ADM are identified and authenticated, and OE.ADMIN, showing that S S.INIT_ADM, S.BKS_ADM, S.CUST_ADM keep their devices secure.

P.SERVICES_EVAL
This policy is implemented by O.SERVICES, providing the services, and O.ADMIN showing that the random number generation can be managed.

P.AUTHORITY_LEVELS
This policy is implemented by O.AUTHORITY_LEVEL, which is a direct translation.

P.AUTHORITY_LEVEL_CHANGE
This policy is implemented by O.AUTHORITY_LEVEL_CHANGE, which is a direct translation.

P.ADMIN
This policy is implemented by:

- O.LOAD_APPLICATION, showing how D.DEP_APPL can be loaded;
- O.LOAD_KEYS, showing how D.APPL_KEYS can be loaded;
- O.BACKUP_KEYS, showing how D.APPL_KEYS can be backed-up and restored;
- O.ERASE, showing how S.DEP_APPL and/or D.APPL_KEYS can be deleted from the TOE;
- O.ADMIN, showing how all of these operations can be enabled. Disabled and/or limited.

8.2. SECURITY REQUIREMENTS RATIONALE

The purpose of the Security Requirements Rationale is to demonstrate that the security requirements are suitable to meet the Security Objectives.

8.2.1. The SFRS meet the Security Objectives for the TOE

For each Security Objective for the TOE we demonstrate that it is met by the SFRs. The tracings are provided implicitly by the rationales.

O.AUTHORITY_LEVEL and O.AUTHORITY_LEVEL_CHANGE
These security objectives are directly implemented in FMT_SMF.1 (showing which operations are allowed) and FMT_MOF.1 (showing who can perform which operations).

O.ADMIN_I&A
This security objective is met by FMT_SMR.1 (showing that there are three roles), FIA_UID.2 and FIA_UAU.2, showing that they need to be identified and authenticated before performing any role-specific actions.

O.PHYSICAL_TAMPER
This security objective is met by FPT_PHP.3 (providing physical resistance), FAU_GEN.1 (detecting and logging tamper events), FPT_STM.1 (providing a reliable time for logging purposes), FAU_SAA.1 (allowing detection of potential violations of the TSP) and FAU_ARP.1 (deleting S.DEP_APPL and D.APPL_KEYS). Finally, FDP_RIP.1 ensures that S.DEP_APPL and in particular D.APPL_KEYS cannot be retrieved after being deleted.

O.LOGICAL_TAMPER
This security objective is directly met by FPT_RVM.1 (preventing bypass) and FPT_SEP.1 (preventing logical tampering with the TSF and the TSF data).

O.SERVICES
This security objective is met by the collection of FCS_RND.1, all FCS_CKM.1+* and all FCS_COP.1+* requirements, showing in detail how each service is provided.

O.LOAD_APPLICATION
This security objective is met by FDP_ACC/ACF.1+1 specifying the rules for loading D.DEP_APPL, FDP_ITC.2 specifying how it is loaded, and FDP_DAU.2 showing that it must be signed by S.BKS.ADM before accepting it.

O.LOAD_KEYS
This security objective is met by FDP_ACC/ACF.1+2 specifying the rules for loading S.APPL_KEYS, FDP_ITC.1 specifying how it is loaded, and FTP_ITC.1 specifying a trusted channel with a C-ZAM/DEP to protect against modification and disclosure.

O.BACKUP_KEYS
This security objective is met by FDP_ACC/ACF.1+2 specifying the rules for backing up and restoring keys and that nobody is allowed to read the backed up keys.

O.ERASE
This security objective is met by FDP_ACC/ACF.1+1 and FDP_ACC/ACF.1+2 specifying the rules for erasing D.APPL_KEYS and/or S.DEP_APPL.

O.ADMIN
This security objective is being met by FDP_ACC/ACF.1+1, FDP_ACC/ACF.1+2, FDP_ACC/ACF.1+3 showing that the security attribute Capability must be LOADED in order to perform the operations, FMT_MSA.1 showing that only S.CUST_ADM can change this attribute to LOADED (possibly with a time-limit (FMT_SAE.1) or use-limit (FMT_SAE.2), and FMT_MSA.3 showing that the security attributes of the operations start out as UNLOADED.

O.STRONG_RANDOM
This security objective is being met by FCS_RND.1 and FCS_CKM.1+*, showing that the keys and random numbers are generated, in conjunction with a refined ATE_FUN.1 (see sections 5.1.7 and 5.2 for more details) showing that the random number mechanism (which also generates the “randomness” for the keys) meets two popular test suites for random numbers.

8.2.2. The security requirements for the IT environment meet the security objectives for the environment

In this section it is shown how all IT security objectives for the environment are being addressed by security requirements for the IT environment. The security objectives that are non-IT are not being addressed.

OE.SIGN_SOFTWARE
This objective is addressed by FMT_MSA.2, showing that the author must carefully check whether the Application will not provide unwanted access to any of the cryptographic keys in D.APPL_KEYS.

OE.RANDOM_TEST
This objective is addressed by FPT_TST.1 showing that a test program will be available that allows users to test whether the random number generating function of the TOE continues to work over time.

8.2.3. The Assurance Requirements and Strength of Function Claim are appropriate

The TOE security assurance requirements are equal to the CC Evaluation Assurance Level EAL3 + ADV_FSP.2.

"EAL3 permits a conscientious developer to gain maximum assurance from positive security engineering at the design stage without substantial alteration of existing sound development practices. EAL3 is applicable in those circumstances where developers or users require a moderate level of independently assured security, and a thorough investigation of the TOE and its development without substantial re-engineering. (CC Part 3 para 209-210)"

This level was found to provide the best cost/assurance benefit by prospective customers for this TOE.

The augmentation of ADV_FSP.2 was chosen to allow an easy extension to EAL4 in a later stage, without having to redo AGD_USR.1, AGD_ADM.1, ADV_HLD.2, parts of ALC_RCR.1, ATE_COV.2, ATE_IND.2.

No strength-of-function claim was made.[78]

8.2.4. All dependencies have been met

The dependencies between SFRs are shown in section 5.1, where it was shown in detail how each dependency was addressed.

The dependencies between SARs are fulfilled because all dependencies in EAL3 are satisfied, and ADV_FSP.2 has the same dependencies as ADV_FSP.1

There are no dependencies between SFRs and SARs in this Security Target.

[78] Instead, an alternative way of testing the random number generator is used, based on the DIEHARD test suite and [F140-2MPRL]. See section 5.1.7 for details.

8.2.5. The requirements are internally consistent

A) The SARs are internally consistent, as they are an EAL, with only a minor augmentation (ADV_FSP.2) and this augmentation does not cause inconsistencies.

B) The SARs and SFRs are completely independent of each other, so there are no inconsistencies between them.

C) The SFRs are internally consistent because:
C1) They have been divided in several, relatively independent groups:

- *Services offered by the TOE*: SFRs for the various services offered by the TOE: cryptographical services, key generation services and random number generation services.
- *Loading Application Software*: SFRs that allow the TOE to load and erase software.
- *Loading, backing up, and erasing Application Keys*
- *Managing the TOE*: SFRs that allow administrators to manage the TOE and control access to its services.
- *Tampering and abnormal operating conditions*: SFRs protecting the TOE against physical tampering, unusual temperatures etc.
- *Architectural security* SFRs that ensure that the TOE cannot be corrupted or bypassed.

C2) Some of these groups are completely independent from other groups. These groups are Services offered by the TOE, Tampering and abnormal operating conditions, and Architectural security. Each group is small enough to see that it is internally consistent. These groups can therefore be safely removed from the SFR consistency analysis.

C3) The remainder was analysed on common subjects, objects and operations that may cause inconsistencies. The following subjects, objects and operations were scrutinised:

- Application Software (D.DEP_APPL and S.DEP_APPL). After analysis of the SFRs no inconsistencies with this Subject/Object could be found,
- Application Keys (D.APPL_KEYS and D.BKUPD_KEYS). After analysis of the SFRs no inconsistencies with this Subject/Object could be found.
- Management Roles (S.INIT_ADM, S.BKS_ADM, and S.CUST_ADM) especially in combination with Authority Levels. After analysis of the SFRs no inconsistencies with this Subject/Object could be found.

The SFRs are therefore considered to be internally consistent.

D) The security requirements for the IT environment are internally consistent and independent of the other requirements.

As the SFRs, SARs and security requirements for the IT environment are all internally consistent, and no inconsistencies can be found between them, the IT security requirements are internally consistent.

8.2.6. The requirements are mutually supportive[79]

1. The requirements meet the security objectives (see section 8.2.1 and 8.2.2).
2. The assurance requirements are appropriate (see section 8.2.3)
3. All dependencies have been met (see section 8.2.4).
4. The requirements are internally consistent (see section 8.2.5)
5. Supporting SFRs were included in the ST: notably FPT_RVM.1 (against bypass), FDP_RIP (to hide residual information), FPT_SEP (against logical tampering), FPT_PHP (against physical tampering), FMT_MSA.1 (to restrict the capability of modifying security attributes) and FMT_MOF/SMF (to restrict the capability of modifying security functions)

The security requirements are therefore considered to be mutually supportive.

[79] This argument has been based on section 8.3.4 of Guide for the production of PPs and STs, PDTR 15446 N2449

8.3. TOE SUMMARY SPECIFICATION RATIONALE

8.3.1. The functions meets the SFRs

For each SFR we demonstrate that it is met by the Security Functions. The tracings are provided implicitly by the rationales.

FCS_CKM.1+1 Cryptographic key generation (for AES)
This SFR is directly implemented by KEY_1.

FCS_COP.1+1 Cryptographic operation (AES)
This SFR is directly implemented by CRYPT_1.

FCS_COP.1+2 Cryptographic operation (AES CBC-MAC)
This SFR is directly implemented by CRYPT_2.

FCS_CKM.1+2 Cryptographic key generation (for DES and 3DES)
This SFR is directly implemented by KEY_2.

FCS_COP.1+3 Cryptographic operation (DES/3DES)
This SFR is directly implemented by CRYPT_3.

FCS_COP.1+4 Cryptographic operation (DES/3DES CBC-MAC)
This SFR is directly implemented by CRYPT_4.

FCS_CKM.1+3 Cryptographic key generation (for RSA)
This SFR is directly implemented by KEY_3.

FCS_COP.1+5 Cryptographic operation (RSA)
This SFR is directly implemented by CRYPT_5.

FCS_COP.1+6 Cryptographic operation (RSA signatures)
This SFR is directly implemented by CRYPT_6.

FCS_COP.1+7 Cryptographic operation (hashing)
This SFR is directly implemented by CRYPT_7.

FCS_RND.1 Quality metrics for random numbers
This SFR is directly implemented by RND_1, KEY_1, KEY_2 and KEY_3. For the relationship between these functions see section 6.1.6.

FDP_ACC.1+1 Subset access control (Applications) and FDP_ACF.1+1 Security attribute based access control (Applications) and FDP_ITC.2 Import of user data with security attributes and FDP_DAU.2 Data authentication with identity of guarantor

- The loading of D.DEP_APPL through the PCI-bridge and rejection if it was not signed by BKS_ADM is implemented by LOAD_1. The fact that LOAD_1 is disabled by default is given in MANAG_2, and that it can be enabled/limited/disabled is given in MANAG_1.
- The erasing of S.DEP_APPL (and D.APPL_KEYS) from the TOE is implemented by ERASE_1. The fact that ERASE_1 is disabled by default is given in MANAG_2, and that it can be enabled/limited/disabled is given in MANAG_1.

FDP_ACC.1+2 Subset access control (Application Keys) and FDP_ACF.1+2 Security attribute based access control (Application Keys) and FDP_ITC.1 Import of user data without security attributes

- The loading of Application Keys by S.CUST_ADM through the trusted channel is implemented in LOAD_2. The fact that LOAD_2 is disabled by default is given in MANAG_2, and that it can be enabled/limited/disabled is given in MANAG_1.
- The backing up of Application Keys is implemented in BACKUP_1. The fact that BACKUP_1 is disabled by default is given in MANAG_2, and that it can be enabled/limited/disabled is given in MANAG_1.
- The restoring of previously backed-up Application Keys is implemented in BACKUP_2. The fact that BACKUP_2 is disabled by default is given in MANAG_2, and that it can be enabled/limited/disabled is given in MANAG_1.
- The erasing of Application Keys is implemented in ERASE_2. The fact that ERASE_2 is disabled by default is given in MANAG_2, and that it can be enabled/limited/disabled is given in MANAG_1.

FTP_ITC.1 Inter-TSF trusted channel

The trusted channel is directly implemented by CZAM_1. The fact that it is used to load Application Keys is implemented in LOAD_2. The fact that it is used to manage the security functions is implemented in MODE_4 and MODE_5. The fact that is used to manage the security attribute Capability is implemented in MANAG_1

FMT_SMR.1 Security roles

This is directly implemented by CZAM_1, showing the roles.

FIA_UID.2 User identification before any action and FIA_UAU.2 User authentication before any action

These requirements are directly implemented by CZAM_1 showing that the administrators can be identified and authenticated before being allowed to do their security-relevant actions.

FMT_MSA.1 Management of security attributes and FMT_SAE.1 Time-limited authorization and FMT_SAE.2 Use-limited authorisation

These three SFRs are directly implemented by MANAG_1, that shows that only S.CUST_ADM can enable, disable, or limit functions.

FDP_ACC.1+3 Subset access control and FDP_ACF.1+3 Security attribute based access control

The fact that R.RND can be performed is implemented in RND_1. The fact that this function can be disabled is implemented in MANAG_1.

FMT_MSA.3 Static attribute initialisation

This SFR is directly implemented by MANAG_2, showing that all the listed functions are disabled by default.

FMT_SMF.1 Specification of Management Functions and FMT_MOF.1 Management of security functions behaviour

These SFRs are implemented by:

- MODE_1 (for MoveToNONE)
- MODE_2 (for AssignCustomer and SetMode)
- MODE_3 (for MoveToINIT)
- MODE_4 (for MoveToBKS). The trusted channel in MODE_4 is used to ensure that only S.INIT_ADM can use MODE_4.
- MODE_5 (for MoveToCUST). The trusted channel in MODE_5 is used to ensure that only S.INIT_ADM can use MODE_5.

FPT_PHP.3 Resistance to physical attack and FAU_GEN.1 Audit data generation and FPT_STM.1 Reliable time stamps

These SFRs are implemented by PROT_1, ALARM_1 and LOG_1, giving the range of tamper events that can be generated and that they are logged with the correct time.

FAU_SAA.1 Potential violation analysis and FAU_ARP.1 Security alarms and FDP_RIP.1 Subset residual information protection

These SFRs are implemented by REACT_1 that moves the TOE back to NONE level (basically resetting it completely). MODE_1 also specifies that the Application Software and Application Keys are irretrievably gone, thus implementing FDP_RIP.

FPT_SEP.1 TSF domain separation

This SFR is implemented by ARCH_1, ARCH_2, and ARCH_3, showing that the TSF, the Application Software, and the Application keys have their own security domain.

FPT_RVM.1 Non-bypassability of the TSP

This SFR is directly implemented by ARCH_4.

8.3.2. The assurance measures meets the SARs

The statement of assurance measures has been presented in the form of a reference to the documents that show that the assurance measures have been met (CC Part 3 paragraph 188). This statement can be found in section 6.2. This section also contains the required tracings.

8.3.3. The SOF-claims for functions meet the SOF-claims for the SFRs

No SOF-claim is made. This section does therefore not apply.

8.3.4. The functions are mutually supportive

(This argument has been based on section 9.3.8 of Guide for the production of PPs and STs, PDTR 15446 N2449)

The amount of detail introduced by the functions over and above the SFRs is relatively small. On the points where additional detail was introduced, especially at those points already specifically discussed in item C3 of section 8.2.5, it was examined whether this introduced problems for mutual support. This was found not to be the case.

8.4. PP CLAIMS RATIONALE

This Security Target TOE does not claim conformance to any Protection Profile. This section is therefore empty.

9. ANNEXES

9.1. GLOSSARY

Abbreviation	Description
AES	Advanced Encryption Standard
ANSI	American National Standards Institute
CC	Common Criteria
CBC	Cipher Block Chaining – a mode of operation of DES or AES
DES	Data Encryption Standard
EAL	Evaluation Assurance Level
ECB	Electronic Code Book - a mode of operation of DES or AES
FIPS	Federal Information Processing Standard
NIST	National Institute of Standards and Technology
SFP	Security Function Policy
SOF	Strength of Function
ST	Security Target
TOE	Target of Evaluation
TSC	TSF Scope of Control
TSF	TOE Security Function

9.2. REFERENCES

[(3)DES]	National Institute of Standards and Technology (NIST), FIPS Publication 46-3: Data Encryption Standard (DES), 25 Oct 1999.
[AES]	National Institute of Standards and Technology (NIST), FIPS Publication 197: Specification for the ADVANCED ENCRYPTION STANDARD (AES), 26 Nov 2001.
[CBC-MAC]	National Institute of Standards and Technology (NIST), FIPS Publication 113: Specifications for Computer Data Authentication, 30 May 1985.
[DIEHARD]	Florida State University, Department of Statistics, G. Marsaglia: Diehard battery of tests of randomness, The Marsaglia random number CDROM, 1995.
[F140-2MPRL]	The MONOBIT, POKER, RUNS and LONGRUNS tests as described in "National Institute of Standards and Technology (NIST), FIPS Publication 140-2: Security Requirements for Cryptographic Modules, 25 May 2001" (prior to Change Notice #2)
[ISO/IEC 9796-1]	International Institute for Standardization, ISO/IEC 9796-1: Information technology – Security techniques – Digital signature schemes giving message recovery, part 1: Mechanisms using redundancy, 1991.
[ISO/IEC 9796-2]	International Institute for Standardization, ISO/IEC 9796-1: Information technology – Security techniques – Digital signature schemes giving message recovery, part 2: Mechanisms using a hash-function, 1997.
[PKCS#1(5)]	RSA Laboratories, PKCS#1 v2.1: RSA Cryptographic Standard, 14 Jun 2002, section 5: Cryptographic Primitives.
[PKCS#1(8)]	RSA Laboratories, PKCS#1 v2.1: RSA Cryptographic Standard, 14 Jun 2002, section 8: Signature Schemes with Appendix.
[MD5]	Internet Activities Board, R.L.Rivest, RFC1321: The MD5 Message-Digest algorithm, 1992.
[SHA]	National Institute of Standards and Technology (NIST), DRAFT FIPS Publication 180-2: Specifications for the Secure Hash Standard, 2001.

9.3. FIPS 140-2 TESTS

Taken from [F140-2MPRL]:

4.9.1 Power-Up Tests

[…]

Statistical random number generator tests. If statistical random number generator tests are required (i.e., depending on the security level), a cryptographic module employing RNGs shall perform the following statistical tests for randomness. A single bit stream of 20,000 consecutive bits of output from each RNG shall be subjected to the following four tests: monobit test, poker test, runs test, and long runs test.

The monobit test

1. Count the number of ones in the 20,000 bit stream. Denote this quantity by X.
2. The test is passed if 9,725 < X < 10,275.

The poker test

1. Divide the 20,000 bit stream into 5,000 consecutive 4 bit segments. Count and store the number of occurrences of the 16 possible 4 bit values. Denote *f(i)* as the number of each 4 bit value *i*, where $0 \le i \le 15$.
2. Evaluate the following:

$$X = (16/5000) * \left(\sum_{i=0}^{15} [f(i)]^2 \right) - 5000$$

3. The test is passed if 2.16 < X < 46.17.

The runs test

1. A run is defined as a maximal sequence of consecutive bits of either all ones or all zeros that is part of the 20,000 bit sample stream. The incidences of runs (for both consecutive zeros and consecutive ones) of all lengths (≥ 1) in the sample stream should be counted and stored.
2. The test is passed if the runs that occur (of lengths 1 through 6) are each within the corresponding interval specified in the table below. This must hold for both the zeros and ones (i.e., all 12 counts must lie in the specified interval). For the purposes of this test, runs of greater than 6 are considered to be of length 6.

Length of Run	Required Interval
1	2,315 – 2,685
2	1,114 – 1,386
3	527 – 723
4	240 – 384
5	103 – 209
6+	103 - 209

Table 3. *Required intervals for length of runs test*

The long runs test

1. A long run is defined to be a run of length 26 or more (of either zeros or ones).
2. On the sample of 20,000 bits, the test is passed if there are no long runs.

9.4. DOCUMENT HISTORY

Version Management Report			
Version	Name(s)	Date	Comments
0.01 to 0.97	debis IT Security Services: - Reinhard Herwig - Thomas Borsch - Peter Klein	up to 19 Oct 2001	Initial drafts (ITSEC) Rework to CC conformance Internal workshops & reviews
0.97.2	TheSteamFactory: - Ronny Op de Beeck	30 Oct 2001	Updates using comments of the banksys DEP team (gathered during the 'Friday morning sessions')
0.97.3		28 Nov 2001	
1.0 (1)		21 Dec 2001	Final Draft: remove FPT_EMSEC, update cross references, simplify document structure, fix the level to EAL3+, complete review.
1.0 (2)		06 Feb 2002	Update after first comments of TNO, complete rewrite 'TOE Security Functions' chapter
1.0 (3)		10 Jun 2002	Update after 2nd comments of TNO
1.0 (4)	TNO-ITSEF BV - Dirk-Jan Out	5 Jul 2002	Update by TNO to harmonise TSS and REQ. Many sections have to be adapted.
1.0 (5)	TNO-ITSEF BV - Dirk-Jan Out	29 Jul 2002	Version for evaluation.
1.0 (6)	TNO-ITSEF BV - Dirk-Jan Out	16 Aug 2002	Update based on: - evaluation comments "IR DEP/PCI v1.0" - BSI comments "Review Protocol #1"
1.1 (1)	TNO-ITSEF BV - Dirk-Jan Out	21 Oct 2002	Update based on - evaluation comments "IR DEP/PCI v2.0"
1.1 (2)	TNO-ITSEF BV - Dirk-Jan Out	26 Feb 2003	Update based on - evaluation comments "IR DEP/PCI v3.0" and comments from BSI
1.1 (3)	TheSteamFactory: - Ronny Op de Beeck	10 Mar 2003	Update of the Assurance Measures
1.1.(4)	TheSteamFactory - Ronny Op de Beeck Banksys - Filip Demaertelaere	08 May 2004	Update of the Assurance Measures Update TOE Guidance components Minor updates FIPS 140-2