Calypso
Networks Association

CALYPSO SPECIFICATION

Calypso Basic - Protection Profile

Certified by

Agence Nationale de la Sécurité des Systèmes d'Information (ANSSI)

under the reference ANSSI-CC-PP-2021/01

Foreword

This document was developed through the Calypso Networks Association Workgroup 1 (specifications and technical development).

This document is the result of a joint work undertaken by the members of the WP12.

The association wishes to thank its active contributors to this document: INFINEON, SPIRTECH.

A special thank you is addressed to INTERNET OF TRUST for its leading role in the edition of this document.

Author

Calypso
Networks Association

Editor

INTERNET OF TRUST

Links and Contacts

Calypso related Internet sites:

Calypso Networks Association	https://www.calypsonet.org
Calypso Technical Support	https://www.calypsostandard.net

Calypso related email contacts:

Calypso Networks Association	contact@calypsonet.org
Calypso Technical Support	support@calypsonet.org

Revision List

Version	Date	Modifications
1.0	26-10-2021	First published version

Table of Contents

1 INTRODUCTION

1.1 Identification

Title	Calypso Basic - Protection Profile
Identification	210311-SP-CalypsoBasic_ProtectionProfile
Version	1.0
Date	26/10/2021
Sponsor	Calypso Networks Association
Technical editor	Internet of Trust
CC Version	3.1 Revision 5
Certification reference	ANSSI-CC-PP-2021/01

1.2 Overview

A Calypso Basic product is a contactless smartcard, i.e. a portable object (PO) with an ISO/IEC 14443 interface, running a single Calypso Basic application. Such products focus on providing access to public transportation and possibly other associated services that can be combined into a transport title or ticket.

A transport title or ticket is used during a ticket validation or control process. The rights written in the Calypso Basic product's files are checked to establish whether the user's entrance to a transport network or access to a delivery of services or goods is allowed.

This document defines the Calypso Basic Protection Profile (PP) which includes the minimum-security requirements for Calypso Basic products, such as secure initialization of the used cryptographic keys, key diversification and a key hierarchy mechanism, secure key storage, access control mechanism to persistent data, secure session, ratification, usage of transaction counters, and memory modification management.

This Protection Profile claims conformance with the assurance package EAL2+ which consists of the predefined EAL2 package augmented with ALC_DVS.1 and AVA_SPECIFIC.1. This is a CC Part 3 extended assurance component that is based on AVA_VAN.2 and which requires resistance to the Enhanced-basic attack potential as defined in [JIL-AAPS].

This document is intended primarily for the use of Calypso Basic product developers, integrators, service providers, as well as for evaluation laboratories, certification bodies, and consumers of Common Criteria certificates. Clarifications and instructions for authors of conformant STs are given in Application Notes all along the document.

This Protection Profile has been developed by the Calypso Networks Association (CNA) and it constitutes the reference for the evaluation of Calypso Basic products in the Common Criteria (CC) Scheme.

1.3 References, Definitions and Acronyms

1.3.1 References

[C-BASIC]	Calypso Specification - Calypso Basic – Version 1.1 (Ref. 191011)
[C-LIGHT]	Calypso Light Application for Portable Objects 'CLAP' – Version 1.2
[C-PRIME]	Calypso Revision 3 Specification – Portable Object Application – Version 3.3
[CC1]	Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model; CCMB-2017-04-001, Version 3.1, Revision 5, April 2017
[CC2]	Common Criteria for Information Technology Security Evaluation, Part 2: Security functional components; CCMB-2017-04-002, Version3.1, Revision5, April 2017
[CC3]	Common Criteria for Information Technology Security Evaluation, Part 3: Security assurance components; CCMB-2017-04-003, Version 3.1, Revision 5, April 2017
[CEM]	Common Methodology for Information Technology Security Evaluation, Evaluation methodology; CCMB- 2017-04-004, Version 3.1, Revision 5, April 2017
[JIL-AAPS]	Application of Attack Potential to Smartcards, Version 3.1, June 2020, edited by JIL
[SOGIS-ACM]	SOG-IS Crypto Evaluation Scheme Agreed Cryptographic Mechanisms, Version 1.2 January 2020
[CEN/TS 16794]	Public transport – Communication between contactless readers and fare media – Part 1: Implementation requirements for ISO/IEC 14443 Part 2: Test plan for ISO/IEC 14443
[EN 1545]	Identification card systems – Surface transport applications – Part 1: Elementary data types, general code lists and general data elements Part 2: Transport and travel payment related data elements and code lists
[ISO/IEC 7816-4]	ISO/IEC 7816-4:2020 Identification cards — Integrated circuit cards — Part 4: organization, security and commands for interchange
[ISO/IEC 9797-1]	ISO/IEC 9797-1:2011 Information technology — Security techniques — Message Authentication Codes (MACs) — Part 1: Mechanisms using a block cipher
[ISO/IEC 14443]	Identification cards – Contactless IC cards – Proximity cards – Part 1: Physical characteristics Part 2: Radio frequency power and signal interface Part 3: Initialization and anticollision Part 4: Transmission protocol

[ISO/IEC 18033-3] ISO/IEC 18033-3:2010 Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers

1.3.2 Definitions

The following meanings apply to SHALL, SHALL NOT, MUST, MUST NOT, SHOULD, SHOULD NOT, and MAY in this document:

- SHALL indicates an absolute requirement, as does MUST.
- SHALL NOT indicates an absolute prohibition, as does MUST NOT.
- SHOULD and SHOULD NOT indicate recommendations.
- MAY indicates an option.

Selected terms used in this document are included below. Additional terms are defined in Common Criteria [CC1].

Term	Definition
Calypso Serial Number (CSN)	Unique serial number given to a Calypso product.
Debit Key	One of the three cryptographic secret keys used by the Calypso Basic application. It is the lowest element in the key hierarchy that is put into place, i.e. all actions allowed with the Debit Key are also allowed with the Load Key and with the Issuer Key.
Integrity	Property that means that data is protected from unauthorized changes. It applies by extension to executable code.
Issuer Key	One of the three cryptographic secret keys used by the Calypso Basic application. It is used as a personalization key and it is the highest element in the key hierarchy that is put into place, i.e. all actions allowed with the Debit Key or the Load Key are also allowed with the Issuer Key but no action requiring the Issuer Key is allowed with any of the other two keys.
Key Identifier	*V*alue identifying the function (KIF) and version (KVC) of the key.
Load Key	One of the three cryptographic secret keys used by the Calypso Basic application. It is used as a reloading key and it is the middle element in the key hierarchy that is put into place, i.e. all actions allowed with the Load Key are also allowed with the Issuer Key and all actions allowed with the Debit Key are also allowed with the Load Key.
Monotonicity	The property of a variable whose value is either always increasing or always decreasing over time. In this document, monotonicity refers to values that are decreasing over time.
Ratification	A security mechanism provided by Calypso Basic products meant to avoid rejecting a legitimate user or debiting a user multiple times if a communication error occurs during the answer to a successful Close Session command.
Secure Session	A process during which the following is simultaneously performed: • the authentication of the Calypso Basic product; • the authentication of the secure module (SAM) used by the terminal; • the authentication of all the data exchanged during the session;

Term	Definition
	• the proof that modifications have been correctly done; • the integrity of the data modification (whole transaction anti-tearing).
Secure Session Mechanism	A security mechanism provided by Calypso Basic products that ensures that all the modifications made during a session are atomic: all modifications are completely and correctly done, or else none of them is done. A session is limited to 3 generic modification commands plus a counter file modification command. The secure session is managed by two commands: • Open Secure Session which indicates the beginning of a secure session; • Close Secure Session which indicates the end of the secure session. A random number is provided by the terminal at session opening. During a successful Open Secure Session command, the Transaction Counter is decreased by one and returned to the terminal with a random number generated by the Calypso Basic application. When closing the session, a MAC (ISO/IEC 9797-1 Algorithm 1 with Padding method 2 using TDES key) is computed on more or less all the exchanged command/response pairs between the Open Session and Close Session commands, using the key indicated during the Open Session command. During the Close Session command, the terminal sends the high part of the MAC to authenticate the exchanged data. The Calypso Basic application answers with the low part of the MAC if the command is successful (i.e. the high part of the MAC sent by the terminal corresponds to the high part of the computed MAC by the Calypso Basic application). All the modifications on the file system are committed only when the Close command is successful; in all other cases (no Close Session commands or a bad MAC, for example), the modifications are cancelled. The complete definition is given in [C-PRIME].

1.3.3 Acronyms

AID	*Application Identifier*
CC	*Common Criteria* (defined in [CC1], [CC2], [CC3])
CEM	*Common Evaluation Methodology* (defined in [CEM])
CNA	*Calypso Networks Association*
CSN	*Calypso Serial Number*
DES	*Data Encryption Standard*
DF	*Dedicated File* (as defined in [ISO/IEC 7816-4])

EAL	*Evaluation Assurance Level* (defined in [CC1])
EF	*Elementary File* (as defined in [ISO/IEC 7816-4])
HSM	*Hardware Security Module*
IC	*Integrated Circuit*
ISO/IEC	*International Organization for Standardization / International Electrotechnical Commission*
MAC	*Message Authentication Code*
MSB	*Most Significant Byte*
NFC	*Near Field Communication*
OSP	*Organizational Security Policy* (defined in [CC1])
PP	*Protection Profile* (defined in [CC1])
PO	*Portable Object*
ROM	*Read-Only Memory*
SAM	*Secure Application Module*
SAR	*Security Assurance Requirement* (defined in [CC1])
SFP	*Security Function Policy* (defined in [CC1])
SFR	*Security Functional Requirement* (defined in [CC1])
SPD	*Security Problem Definition* (defined in [CC1])
ST	*Security Target* (defined in [CC1])
TDES	*Triple-DES* or *3DES* (defined in ISO/IEC 18033-3)
TOE	*Target of Evaluation* (defined in [CC1])
TSF	*TOE Security Functionality* (defined in [CC1])

2 TOE OVERVIEW

2.1 TOE Type

This sub-chapter defines the type of the Target of Evaluation (TOE).

The TOE type is a Calypso Basic product in a contactless integrated circuit (IC) with embedded software, including a single Calypso Basic application that is functionally compliant with the Calypso Basic specification [C-BASIC]. The Calypso Basic product has an ISO/IEC 14443 interface as defined in [C-BASIC].

Calypso Basic products are tailored for thin cards.

The TOE either does not implement testing or debug interfaces or these are irreversibly disabled in the end-user phase (also called production mode).

The TOE comprises:

- All hardware, firmware, and software used to provide the security functionality of the Calypso Basic product;
- The guidance for the secure usage of the Calypso Basic product after delivery.

The TOE does not comprise:

- The SAM/HSM Secure Modules that are used for key storage, authentication between a Calypso Basic product and a terminal, and for performing all cryptographic computations needed to manage the Calypso Basic commands, i.e. key derivation, MAC computation, etc.

In the Calypso Basic specification [C-BASIC], the term portable object (PO) is used for any type of Calypso product (e.g. contactless smartcards, mobile phones with contactless communication (NFC), wristwatches with an embedded contactless component, etc.). In the rest of the document Calypso Basic product and portable object are used interchangeably.

2.2 TOE Description

A Calypso Basic product is a part of the same information system architecture as a Calypso Prime portable object [C-PRIME], a Calypso HCE application, or a Calypso Light product and it follows the same principles.

A Calypso Basic product is a contactless integrated circuit with embedded software supporting the initialization (Phase 2) and the pre-personalization (Phase 3) of the TOE, and containing the Calypso Basic application.

The data of a Calypso Basic application is organized in hierarchical files as defined in ISO/IEC 7816-4. There are two main types of files: Elementary Files (EFs) and Dedicated Files (DFs), i.e. directories which may contain EFs and other DFs. A Calypso Basic product contains a single application which is called the Calypso Basic DF.

A Calypso Basic product:

- Complies with ISO/IEC 14443 and CEN/TS 16794;

- Complies with ISO/IEC 7816-4;
- Allows management of any data, for example EN 1545 transport data structures;
- Ensures the functional security of transactions;
- Ensures fast contactless transactions.

ISO/IEC 14443 and CEN/TS 16794 Compliance:

- A Calypso Basic product is compliant with the ISO/IEC 14443 standard, including parts 1, 2, 3 and 4, and to CEN/TS 16794. No other low-level (contactless or contact) protocol may be managed by the portable object.

ISO/IEC 7816-4 Compliance:

- Calypso Basic data are organized in files, according to the ISO/IEC 7816-4 standard. The file structure is fixed by Calypso specifications.

EN 1545 Transport Data Structures:

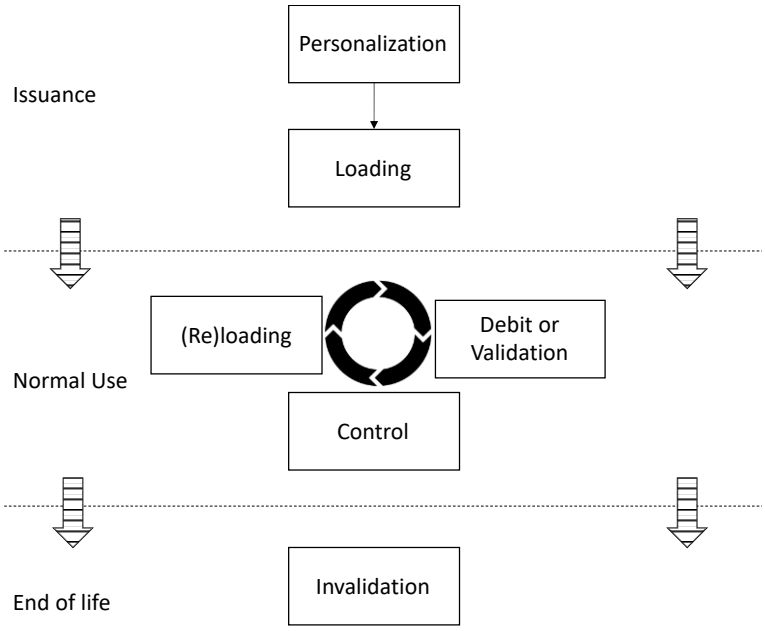
- For public transport applications, the Calypso Basic files may encode EN 1545 transport data structures. Such an encoding is out of the scope of this document and is therefore not described.

  Application Note: Calypso Basic products never analyze the data stored in their files. It may thus be coded differently for each application. It is entirely up to terminals using the portable objects (validation or reloading machines, etc.) to decide the meaning of the data written in the applications.

The technical specificities of the considered Calypso Basic products include the following:

- Only a contactless interface is available (contact mode is not available);
- Only one fixed file structure is available (file structures are not configurable).

The TOE provides functionalities that can be divided into two different categories, namely core end-user functionality and administrative functionality.



The core end-user functionality includes:

- A (re)loading process, i.e. acquiring (access) rights to a transport network or service and loading these on a Calypso Basic product; the Load Key must be used for this operation;
- A debit transaction process, i.e. validating the user's right to enter/gain access to a network or service, recording the transaction and updating the rights accordingly using the Debit Key.

The administrative functionality includes:

- A personalization process, i.e. using the Issuer Key, data may be initialized as necessary;
- An invalidation process, i.e. the user's rights loaded on a Calypso Basic product are invalidated; any of the three keys may be used to this end, including the Debit Key.

2.3 TOE Major Security Features

Calypso Basic products ensure the security of ticketing transactions by providing authentication between the portable object and the terminal. To avoid brute force attacks, a low transaction counter value is used.

The access to data in a Calypso Basic application follows a system of access conditions specific to each file. Some of these access conditions (e.g. 'Session') impose cryptographic computations using secret keys stored in the Calypso Basic product.

The major security features of the TOE are the following:

- Usage of cryptographic algorithms based on TDES Secret Keys
    - A Calypso Basic application will contain the following cryptographic secret keys (the application keys):
        - Issuer Key – used as a personalization key;
        - Load Key – used as a reloading key;
        - Debit Key – used as a debit key;
    - A key hierarchy is put into place:
        - All actions allowed with the Debit Key are also allowed with the Load Key and with the Issuer Key;
        - All actions allowed with the Load Key are also allowed with the Issuer Key;
- Access control mechanism: each file is associated to its access conditions and there exists an access mode and key index for each group of commands. The Calypso Basic file structures includes:
    - The DF – the Dedicated File of the Calypso Basic application and the following elementary files (EFs):
        - An Environment file;
        - An Events Log file;
        - A Contract List file;
        - A Contracts file;
        - A Counters file.

- Secure session mechanism;
- Ratification mechanism;
- Usage of a Transaction Counter that is decreased at each session opening;
- Memory modification management: writing atomicity.

All the cryptographic keys stored in a Calypso Basic product are fully defined and diversified: the key values of each Calypso Basic product are derived from master keys and each derived key is computed by a cryptographic operation using the Calypso Basic Serial Number. The key values for each Calypso Basic product are different and unique so that if the cryptographic keys of one product are compromised, the keys of other products are not directly compromised.

To ensure the uniqueness of a session a challenge is generated during the initial exchanges between a terminal and a card. This challenge not only makes part of the MAC to be calculated but it also serves as a diversifier to create the session key. One of the elements present in the part of the challenge provided by the card is the Transaction Counter thus ensuring that no challenge will ever be repeated. Another element is a random number, thus ensuring the unpredictability of the challenge.

Except for the derivation of session keys from application keys for use in a secure session, no additional diversification or definition process is required at the Calypso Basic product level during transactions.

2.4 TOE Usage

The main use cases for the TOE stem from the electronic ticketing sphere and focus essentially on access to public transportation and the ensuing contactless ticketing transactions between Calypso Basic products and terminals.

In some cases, a Calypso Basic product may be used with other services in combination with transport. Some examples are the use of a tourist card that provides both access to the transport networks as well as specific museums or a park and ride contract that provides not only access to the transport system but also allows entry/exit to a closed park. All these use cases can be performed in the mono-application environment that comes with a Calypso Basic product with all actors having access to the same files and data inside the product.

In general, user contracts are written on the Calypso Basic product establishing the user's right to enter a network or make use of certain services or goods. Calypso Basic products interact and are used in conjunction with terminals, i.e. vending and/or validation machines.

The TOE usage follows most of the typical transaction examples described in [C-LIGHT].

2.5 Available Non-TOE Hardware/Software/Firmware

The TOE may require some non-TOE Hardware, Software or Firmware. However, the TOE must be realized in such a manner that any of the TOE security functionalities do not rely on the proper behavior of non-TOE Hardware, Software or Firmware.

Application Note: Security Targets conformant to this PP shall provide a complete description of the available non-TOE hardware/software/firmware with the list of non-TOE resources used by the TOE.

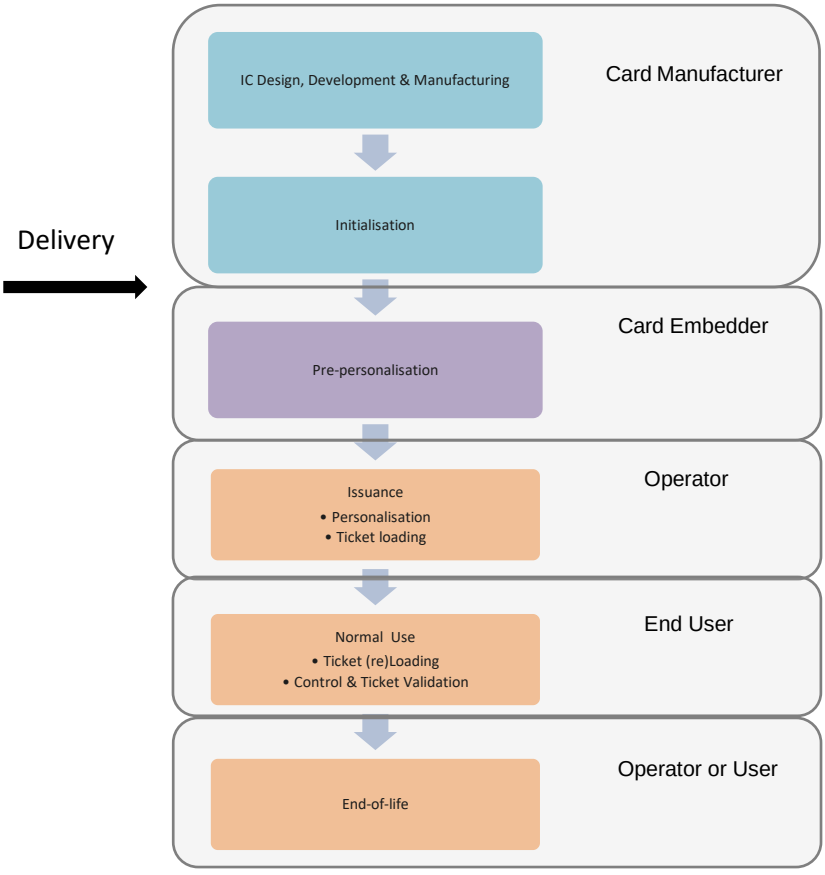
Application Note: Attention is drawn to the fact that the Calypso Basic specification [C-BASIC] stipulates that after entering Phase 2, a Calypso Basic product shall have one Calypso Basic application and no other feature, except potentially, for features related only to the technical management of the product which are not in contradiction with any requirement of the Calypso specification.

2.6 TOE Life Cycle

The product life cycle outlined in this sub-chapter is a reference life cycle that can be instantiated according to development, manufacturing and assembly processes. It is split in six phases as follows:

- Phase 1 corresponds to the IC and embedded software design, development, and manufacturing;
- Phase 2 corresponds to initialization;
- Phase 3 corresponds to pre-personalization;
- Phase 4 covers the issuance phase;
- Phase 5 consists in the product's normal use, namely ticket loading, ticket validation and control;
- Phase 6 stands for the product's end-of-life.

The diagram below summarises the phases of the reference life cycle and indicates the authority operating at each.



The first two phases are performed by or on-behalf of the chip manufacturer.

In Phase 1, the configuration of the Calypso Basic product, including the Calypso Basic File structure is written (and hard-coded).

In Phase 2, the chip's ID (consisting of information such as the unique identification number and optionally, of manufacturing information such as date of activation, year of manufacturing, etc.) and the serial number are written by the chip manufacturer.

Phase 3 is performed by the card embedder and consists in writing the start-up information (including the AID and the three Application keys) using ISO/IEC 7816-4 commands and pre-defined commands.

Phases 4 covers the issuance phase that consists in personalising the Calypso Basic product for the end-user and optionally loading the initial contract. This is performed by the product operator.

Phase 5 stands for the normal end-usage of the product and is done at the ticketing network level. During a ticket (re)loading, new rights are written into the product's files.

Phase 6 represents the end-of-life for the product and may consist in its invalidation or in its physical destruction.

The TOE delivery point establishes the limits of the evaluation and is located after the end of Phase 2. It must necessarily follow the writing of the chip's ID and the setting of the Calypso Basic Serial Number. The delivery point cannot be moved-up.

Application Note: The file attributes are written during initialization, at Phase 2 of the life cycle. They become non-modifiable afterwards.

Application Note: TOE samples with test keys must be provided for testing or the evaluator should have the capability to load the application keys.

Application Note: The phases into which the life cycle is split and described in this document map to those described in the Calypso Basic specification [C-BASIC] as follows:

- Phase 1 as defined here is newly introduced;
- Phase 2 and Phase 3 as defined here (i.e. initialization and pre-personalization) corresponds to Phase 1 as defined in [C-BASIC];
- Phase 4 as defined here (issuance phase) corresponds to the ensemble of Phase 2 and Phase 3 as defined in [C-BASIC];
- Phase 5, i.e. the product's normal use as defined here corresponds to Phase 4 as defined in [C-BASIC];
- Finally, Phase 6 as defined here, i.e. the product's end-of-life corresponds to Phase 5 as defined in [C-BASIC].

Application Note: Security Targets conformant to this PP must provide a complete description of the life cycle implemented for the TOE and provide the precise data and/or keys that are injected in each phase.

3 CONFORMANCE CLAIMS

3.1 Conformance Claim to CC

This Protection Profile is conformant to the Common Criteria version 3.1 Revision 5, i.e. CC Part 1 [CC1], CC Part 2 [CC2] and CC Part 3 [CC3].

It claims CC Part 2-extended conformance and CC Part 3-extended conformance.

CC Part 2 is extended with FCS_RNG 'Generation of random numbers'.

CC Part 3 is extended with the security assurance component AVA_SPECIFIC.1 'Vulnerability analysis of Calypso Basic products' at Enhanced-basic attack potential.

The evaluation is driven by the following documents:

- Common Methodology for Information Technology Security Evaluation, Evaluation methodology [CEM];
- Application of Attack Potential to Smartcards [JIL-AAPS].

3.2 Conformance Claim to a PP

This PP does not claim conformance to any other PP.

3.3 Conformance Claim to a Package

This PP is conformant to the EAL2+ assurance package which consists of the predefined EAL2 package augmented with ALC_DVS.1 and AVA_SPECIFIC.1, as defined in section 6.

3.4 Conformance Statement

The conformance to this PP, required for the Security Targets and Protection Profiles claiming conformance to it, is strict, as defined in CC Part 1 [CC1].

4 SECURITY PROBLEM DEFINITION

4.1 Assets

This section presents the assets of the TOE and their properties: integrity, confidentiality, monotonicity, and atomicity. Note that integrity implies protection against rollback.

4.1.1 Serial Number

The Calypso Basic Serial Number is assigned during the initialization phase, i.e. Phase 2, and is globally unique among all Calypso Basic products and Calypso applications.

Properties:

- The serial number shall be protected in integrity.

Application Note: The serial number is an 8-byte long number with 2 MSB equal to 0000h and the two most significant bits of the 4th byte set to 01b.

Application Note: The serial number is written during initialization in Phase 2 and becomes non-modifiable afterwards during the lifetime of the portable object. The serial number is unique.

4.1.2 Application Keys

The secret cryptographic application static keys on which the Calypso Basic application relies and which are loaded in Phase 3, i.e. after TOE delivery.

A Calypso Basic application relies on three static TDES keys, namely:

- The Issuer Key, used as a personalization key (key #1) and allowing authenticating the value of any file data and modification of the data of files requiring any of the three keys;
- The Load Key, used as a reloading key (key #2) and allowing authenticating the value of any file data and modification of the data of files requiring the Load Key or the Debit Key;
- The Debit Key, used as a validation key (key #3) and allowing authenticating the value of any file data and modification of the data of files requiring the Debit Key.

A static key is identified by a public parameter composed of the key function and key version.

Session keys are derived from static keys for use in a secure session.

Properties:

- The static key identifiers shall be protected in integrity.
- The static application keys shall be protected in confidentiality.
- The static application keys shall be protected in integrity (non-modifiable).
- The session key derivation mechanism and by extension the session keys shall be protected in integrity.
- The session keys shall be protected in confidentiality.

Application Note: The static application keys should be unique for each portable object. They are derived from master keys and are computed by a cryptographic operation using the Calypso Basic Serial Number. This diversification occurs before their loading into the Calypso Basic product at Phase 3. The static application keys and the key identifier become non-modifiable from Phase 4.

Application Note: In order to choose the secret key that must be used, terminals require the key identifier and the Calypso Basic Serial Number. The latter is received as a response to a select_application() command. The open_secure_session() command returns the former.

4.1.3 File Contents

The contents of any file in the file structure of the Calypso Basic application, including files containing the user data or contracts loaded during a reloading operation as well as the Calypso Basic DF.

Properties:

- The file contents shall be protected in integrity.

Application Note: The Calypso Basic application provides a fixed file structure that manages:

- Two types of ISO/IEC 7816-4 files, organized in records of 29 bytes:
  - EF Linear files
  - EF Cyclic files
- One type of file defined in the Calypso Basic specification [C-BASIC], organized in 1 record of 3 bytes that is managed using proprietary commands (for example Decrease command):
  - EF Counter files, the counter is a 24-bit unsigned big-endian number.

Remark: A Calypso Basic product contains only one DF which is under the MF if it exists (but not under another DF) and is not the MF itself.

4.1.4 File Attributes

The file attributes (e.g. number of records per file, type of records, etc.) including the access conditions of all the files in the Calypso Basic product file structure.

The access conditions associated to each file:

- The access mode (Always/Never/Session) applying for access commands;
- A key index indicating the maximum key index accepted to grant access:
  - Key #1 is also accepted when Key #2 or Key #3 is required for accessing a file
  - Key #2 is also accepted when Key #3 is required for accessing a file.

Properties:

- The file attributes including the access conditions shall be protected in integrity.

4.1.5 DF Status

A value indicating the validity state of the Calypso Basic application (or DF validity); this value is changed only through the Invalidate command.

Properties:

- The DF status shall be protected in integrity.

Application Note: The DF status is coded using one byte:

- 00h: the DF is valid (default value given during initialization, in Phase 2);
- 01h: the DF is definitively invalid. This value marks the product's Phase 6, i.e. the end of its life cycle.

This invalidation is performed by using the Invalidate command and is irreversible. When invalidated, no file in the Calypso Basic application can be modified anymore, regardless of the access conditions.

4.1.6 Transaction Counter

A counter that is stored in non-volatile memory and decremented at each session opening.

Properties:

- The Transaction Counter shall be protected in integrity and it shall be strictly monotonic.

Application Note: When the Transaction Counter reaches 0, the application rejects any requests for opening sessions. For Calypso Basic products the transaction counter is initially set to 1000.

4.1.7 Secure Session – Random Number

A random number generated by the Calypso Basic application and sent to the terminal for identifying an open secure session.

Properties:

- The random number shall be protected in integrity;
- The random number shall meet a well-defined quality metric.

Application Note: The random number is 8 bits long and is generated for each newly opened secure session.

4.1.8 Secure Session – Ratification Status

A persistent flag used for signaling if a terminal successfully received the close secure session answer.

Properties:

- The ratification status shall be protected in integrity.

Application Note: The ratification status will have one of the following values:

- 'not_ratified' which can only be set by the Close Secure Session command;
- 'ratified' which can be set by the Close Secure Session command itself, or by any command received after a Close Secure Session that set the flag to 'not_ratified' and before the deselection of the application. The ratification status is initially set to 'ratified'.

4.2 Users

The users of the TOE are operators and end-users. Both interact with the TOE through terminals.

4.3 Threats

This section introduces the threats against Calypso Basic products.

Threats against the TOE can be classified into technological threats and non-technological threats, such as theft or malevolent physical destruction. The latter are considered out of scope and are not covered by this section.

Technological threats have an impact on either Calypso network operators, end-users of Calypso Basic products or both. They can be achieved by using hardware attacks or software attacks. This PP considers hardware-based and software-based attacks that require Enhanced-basic attack potential (see AVA_SPECIFIC.1).

Hardware attacks include but are not limited to:

- Perturbation attacks (fault injection) which aim to disturb the normal application execution and to produce unexpected behaviors (e.g. extracting some keys using fault analysis during cryptographic operations, modifying a value read in memory (volatile or not), modifying the control flow of the application to bypass a security verification, etc.). These attacks can be used for bypassing security mechanisms such as the secure Session MAC mechanism (e.g. glitch during Session MAC). For instance, a glitch performed during Session MAC verification would allow an attacker to perform reloading without having the necessary keys;
- Tearing attacks, which appear when a power cut occurs before the end of a transaction, and which could allow an attacker to take advantage of partially writing in memory (when using non-atomic operations);
- Side-channel attacks, which target observable physical variations (power consumption, time, electromagnetic emissions, etc.) during operations, in order to retrieve secret data;
- Exploiting unprotected debug or test interfaces, which consist in directly accessing and reading or modifying memory contents by means of a hardware debugging facility. This could lead to an off the shelf attack relying on specifically developed tools.

Software attacks include but are not limited to:

- MITM (man-in-the-middle) or replay attacks, consisting in an attacker hiding in the communication path between the application and the validator and managing to observe and intercept the exchanges, or to modify the exchanges by, for example, adding some commands or sending invalid commands/parameters;
- Direct protocol attacks, which occur when an attacker sends commands in an unexpected order (attacks focusing on the state machine of the application) or invalid commands/parameters;
- Exploitation of overflows and/or other implementation errors such as buffer overflows, integer errors, timing attacks.

Application Note: The technological threats include relay attacks consisting in intercepting authentication attempts between a terminal and a Calypso Basic product, relaying them and thereby gaining unauthorized access to the transport network or services. However, in the context of Calypso ticketing, relay attacks are considered to be a residual risk since such attacks are too complex to implement w.r.t. their gain (granting one access) and therefore, they are not considered in the present PP. It shall be noticed that some inherent properties of the product such as the transaction timing limit this risk and that some additional external (back-end) controls may be added to reduce it further.

4.3.1 T.CLONE_PO: Cloning of a Calypso Basic product

An attacker manages to copy the keys and all other necessary data and/or code of a Calypso Basic product to another device, making the latter accept these keys and code and/or data as genuine.

Assets threatened directly: All.

4.3.2 T.REPLAY_TRANSACTION: Replay of a valid (re)loading or debit transaction

An attacker manages to replay a valid (re)loading or debit transaction.

Assets threatened directly: all (integrity, confidentiality).

Application Note: This threat may require other preliminary steps such as unauthorized manipulation of TC and random number and secret keys recovery. These are presented as individual threats below.

4.3.3 T.MODIFY_FILE: Rollback/Modification of files (fraudulent reload or rights destruction)

An attacker manages to modify data written in the files, e.g. the user contracts or associated data. The attacker's goal is either to increase the user rights contained in the loaded contract, i.e. to increase the value of or to reload some tickets, or to put the end-user at a disadvantage, for instance, by removing all its tickets, contracts or rights through the modification or deletion of the associated files.

Assets threatened directly: File contents (integrity).

4.3.4 T.MODIFY_TC: Rollback/Modification of the Transaction Counter

An attacker manages to modify the value of the Transaction Counter or to perform a rollback to a previous value of the Transaction Counter.

Assets threatened directly: Transaction Counter (integrity, monotonicity).

Application Note: The Transaction Counter is used to limit the use of cryptographic keys and to ensure the session uniqueness. Setting the Transaction Counter to a value greater than the upper limit (e.g. 1000) or rollback of the Transaction Counter would invalidate these two properties.

This threat may be a preliminary step for other attacks such as replay attack.

4.3.5 T.MODIFY_RANDOM: Manipulate random number

An attacker either forces the output of a partially/totally predefined value or predicts the random number to be used in a secure session.

Assets threatened directly: random number and the secure session mechanism.

Application Note: This threat may be a preliminary step for other attacks such as replay attack.

4.3.6 T.MODIFY_ DATA: Modification of the file attributes

An attacker manages to modify the file attributes or any non-modifiable data (integrity) in a non-intended life cycle phase.

Assets threatened directly: File attributes (integrity), non-modifiable data (integrity).

Application Note: This threat may be a preliminary step for other threats such as file contents modification.

4.3.7 T.SWITCH_FS_STATUS: Revalidate or invalidate the filesystem

An attacker manages to switch the filesystem status in order to revalidate or invalidate the Calypso Basic application. The consequences of such unauthorized switches are:

- for revalidation, the attacker is able to perform unauthorized modifications of the filesystem, for instance to increase a value, to modify or reload a contract;
- for invalidation, the end-user is put at a disadvantage, losing any contracts or rights loaded on the fraudulently invalidated product.

Assets threatened directly: DF status (integrity).

4.3.8 T.EXTRACT_SECRET_KEYS: Key extraction

An attacker manages to extract the secret cryptographic keys that are fundamental for the security of the Calypso Basic product.

Assets threatened directly: Cryptographic TDES keys (confidentiality).

Application Note: This attack may be a preliminary step for a cloning attack relying on the extraction of one or several application keys and the duplication of the Calypso Basic Serial Number.

4.3.9 T.ABUSE_OPERATION

An attacker manages to enter an abnormal operation mode or phase to compromise the behavior of the Calypso Basic application (bypass, deactivate or change the security mechanisms) or to modify or extract data. For example, an attacker manages to (re)activate and misuse a test or debug mode or to enter in a non-intended life cycle phase of the product (e.g. transition from Phase 5 to Phase 3/Phase 2 and then be able to perform unauthorized operations such as modification of application keys/file attributes, etc.). Another example consists in an attacker that manages to alter the output of the session key generation.

Assets threatened directly: all (integrity, confidentiality).

4.3.10 T.FORCE_UNSAFE_STATE

An attacker forces the system into an unsafe state forcing it to bypass its security mechanisms such as ratification and the secure session mechanism.

Assets threatened directly: all (integrity, confidentiality).

4.4 Organizational Security Policies

The following policies apply to any Calypso Basic product where the application keys are managed and injected after the delivery point, i.e. from Phase 3. Otherwise, the key management is enforced by the chip manufacturer before delivery and is covered by the evaluation.

4.4.1 OSP.SECRETS

Generation, storage, distribution, destruction, or injection of secret data in the TOE shall enforce the integrity and confidentiality of these data.

This applies to secret data injected before the end-usage phase (the application keys).

4.4.2 OSP.DIVERSIFICATION

The cryptographic keys shall be diversified and unique for each Calypso Basic product. They shall be written once during the initialization phase of the Calypso Basic product.

4.5 Assumptions

This section states the assumptions that apply to the TOE operational environment and its actors.

4.5.1 A.PROTECTION_AFTER_DELIVERY

It is assumed that the TOE and its assets are protected by the operational environment from delivery up to Phase 5, i.e. before entering the end-usage phase.

It is assumed that the personnel using the TOE in the operational environment in Phases 3 and 4 have the required skills to understand and apply the security guidelines for the pre-personalization and personalization of the product and that these guidelines are effectively applied.

5 SECURITY OBJECTIVES

5.1 Security Objectives for the TOE

This section states the security objectives for the TOE.

5.1.1 O.SECURE_SESSION

The TOE shall provide a secure session mechanism for all (re)loading and debit transactions that ensures the mutual authentication between the terminal and the TOE, avoids transaction replay (based on the monotonic Transaction Counter and random data provided by the TOE and the terminal), and performs rollback of all file and any persistent data modifications upon secure session failure.

The TOE shall ensure that the number of attempted successful or failed secure sessions does not exceed a given threshold given by the Transaction Counter.

Application Note: The Calypso Basic specification [C-BASIC] states the following requirements:

- The mutual authentication relies on a derived session key that is unique for each secure session and on the MAC of the sequence of commands that were exchanged during the secure session (maximum 325 bytes).
- A secure session is limited to 3 generic file modification commands plus a counter file modification command.

- The secure session mechanism commits the modifications to the file system only after a successful MAC verification; otherwise, all modifications are cancelled and the TOE's data and/or Application invalidation state remain unchanged.
- The Transaction Counter is strictly monotonic and therefore not rollbacked upon transaction failure.

Application Note: The objectives O.MONOTONIC_TC, O.RNG, O.SESSION_KEYS and O.SESSION_AUTHENTICATION contribute to the enforcement of the objective O.SECURE_SESSION, especially the replay protection.

5.1.2 O.MONOTONIC_TC

The transaction counter shall be decremented for every successful opening of the secure session and never be incremented or rollbacked.

5.1.3 O.RNG

The TOE shall provide a random number generator. Random numbers shall have sufficient entropy.

5.1.4 O.SESSION_KEYS

The TOE shall provide a session key derivation operation of the static application keys (Issuer/Load/Debit) that is compliant with Calypso Basic specification [C-BASIC].

Application Note: The Calypso Basic specification [C-BASIC] ensures the uniqueness of the session keys.

5.1.5 O.SESSION_AUTHENTICATION

The TOE shall provide a session authentication mechanism using the session key that is compliant with the Calypso Basic specification [C-BASIC] .

Application Note: The Calypso Basic specification [C-BASIC] mandates the use of a MAC for performing this authentication and ensures the uniqueness of the MAC.

5.1.6 O.FILE_ACCESS_CONTROL

The TOE shall enforce the read/modify file access conditions: ‘undefined’, ‘Never’, ‘Always’ or ‘Session_i’ where ‘i’ is the minimum required key index for files that are accessible within a secure session only.

Application Note: ‘undefined’ is used to cover the irrelevant cases, i.e. a modification operation that is not defined in the specification. The Calypso Basic specification [C-BASIC] states that for all modification attempts to the file system, the file access conditions are ‘Never’, ‘Always’ or ‘Session_i’, and that for all read attempts to the file system, the file access conditions are ‘Always’ or ‘Never’.

5.1.7 O.FS_STATUS

The TOE shall provide a means to permanently disallow the modification of the file system irrespective of the file access conditions.

The TOE shall ensure that this operation is performed by authenticated users only and that the resulting invalidated state is irreversible.

Application Note: This is achieved through the invalidation command. The DF status reflects the invalidated state.

5.1.8 O.NON-MODIFIABLE_ DATA

The TOE shall ensure that the Calypso Basic Serial Number, the static application keys and key identifiers, the file structure and the file attributes (including access conditions) are not modifiable in non-intended phases and never modified in the end-user phase. More precisely such data is set during the following phases and shall not be modified afterwards:

- The Calypso Basic Serial Number is set during initialization, in Phase 2;
- The static application keys and key identifiers are set during pre-personalization, in Phase 3;
- The file structure and file attributes are set during initialization, in Phase 2.

The TOE shall not provide means to modify such data in non-intended phases.

5.1.9 O.CONFIDENTIAL_KEYS

The TOE shall ensure that the cryptographic static and session application keys are protected against disclosure when stored in non-volatile memory and during processing.

The TOE shall not provide any means to extract or export static or session application keys.

5.1.10 O.CORRECT_OPERATION

The TOE shall ensure the integrity of the execution and shall preserve a secure state upon failure. A failed command or operation shall not expose, leak or corrupt the sensitive data of the TOE.

The TOE shall not provide any means to execute in an unexpected mode, to execute arbitrary code, and/or to perform unauthorized read/write operations, e.g. by unduly (re)activating a test or debug mode.

5.2 Security Objectives for the Operational Environment

The following security objectives apply to the Calypso Basic product operational environment.

5.2.1 OE.SECRETS

Management of secret data (e.g. generation, storage, distribution, destruction, or injection of secret data such as the application keys) performed outside the TOE shall enforce the integrity and confidentiality of these data.

5.2.2 OE.DIVERSIFICATION

The cryptographic keys shall be written once during Phase 3 and they shall be diversified (based on the application Serial Number) and unique for each Calypso Basic product.

5.2.3 OE.PROTECTION_AFTER_DELIVERY

The TOE and its assets shall be protected by the operational environment after delivery and before entering the end-usage phase.

The personnel using the TOE in the operational environment shall have the required skills to understand and apply the security guidelines for the pre-personalization and personalization of the product and these guidelines are effectively applied.

5.3 Security Objectives Rationale

5.3.1 SPD and Security Objectives

The following table provides an overview of the security objectives coverage (TOE and its environment). It shows that all threats and OSPs are addressed by the security objectives. It also shows that all assumptions are addressed by the security objectives for the TOE environment.

SPD \ Objectives	O.SECURE_SESSION	O.MONOTONIC_TC	O.RNG	O.SESSION_KEYS	O.SESSION_AUTHENTICATION	O.FILE_ACCESS_CONTROL	O.FS_STATUS	O.NON-MODIFIABLE_DATA	O.CONFIDENTIAL_KEYS	O.CORRECT_OPERATION	OE.SECRETS	OE.DIVERSIFICATION	OE.PROTECTION_AFTER_DELIVERY
T.CLONE_PO									x	x			
T.REPLAY_TRANSACTION	x	x	x	x	x				x				
T.MODIFY_FILE						x			x				
T.MODIFY_TC		x								x			
T.MODIFY_RANDOM			x							x			
T.MODIFY_DATA								x		x			
T.SWITCH _FS_STATUS							x			x			
T.EXTRACT_SECRET_KEYS									x	x			
T.ABUSE_OPERATION										x			
T.FORCE_UNSAFE_STATE										x			
OSP.SECRETS											x		
OSP.DIVERSIFICATION												x	
A.PROTECTION_AFTER_DELIVERY													x

5.3.2 Threats

The rationale of the coverage of the threats by the security objectives is the following:

T.CLONE_PO The combination of the following objectives ensures protection against the cloning of a Calypso Basic product:

- O.CONFIDENTIAL_KEYS ensures that the cryptographic static keys are protected against disclosure;
- O.CORRECT_OPERATION ensures the integrity of the executed code and protects against exposing of sensitive data.

T.REPLAY_TRANSACTION The combination of the following objectives ensures protection against the replay of a valid (re)loading or debit transaction:

- O.SECURE_SESSION ensures that all transactions take place inside a secure session and that transaction replay is avoided (this is a consequence of the session key derivation mechanism, cf. O.SESSION_KEYS, and MAC calculation, cf. O.SESSION_AUTHENTICATION);

- O.MONOTONIC_TC ensures that the Transaction Counter is strictly monotonic, thus ensuring the uniqueness of the challenge used for the session key diversification;
- O.RNG ensures that the generated random numbers are not predictable, thus ensuring that the challenge used for used for the session key diversification is unpredictable;
- O.SESSION_KEYS ensures that the session key is derived using a secure operation which ensures uniqueness;
- O.SESSION_AUTHENTICATION ensures that each transaction (sequence of commands) is authenticated by a MAC which ensures the uniqueness of the transaction to prevent replay attacks;
- O.CONFIDENTIAL_KEYS ensures that the session key is protected against disclosure.

T.MODIFY_FILE The combination of the following objectives ensures protection against the fraudulent rollback or modification of files:

- O.FILE_ACCESS_CONTROL enforces the write access conditions;
- O.CONFIDENTIAL_KEYS ensures the confidentiality of cryptographic keys, including session keys that may be necessary for modifying files.

T.MODIFY_TC The combination of the following objectives ensures protection against the rollback or modification of the Transaction Counter:

- O.MONOTONIC_TC ensures that the Transaction Counter is protected against incrementation and rollback;
- O.CORRECT_OPERATION ensures the integrity of the execution and prevents unauthorized modifications.

T.MODIFY_RANDOM The combination of the following objectives ensures protection against the modification of the random numbers used inside a secure session:

- O.RNG ensures that a random number generator is used;
- O.CORRECT_OPERATION ensures the integrity of the execution and prevents unauthorized modifications.

T.MODIFY_DATA The combination of the following objectives ensures protection against the modification of the file attributes:

- O.NON-MODIFIABLE_DATA ensures that the file attributes are non-modifiable in non-intended phases and in the end-user phase;
- O.CORRECT_OPERATION ensures the integrity of the execution and prevents unauthorized modifications.

T.SWITCH_FS_STATUS The combination of the following objectives ensures protection against the unauthorized invalidation of the application:

- O.FS_STATUS ensures that the invalidation of the application is performed by authenticated users only;
- O.CORRECT_OPERATION ensures the integrity of the execution and prevents unauthorized modifications and data corruption.

T.EXTRACT_SECRET_KEYS The combination of the following objectives ensures protection against the extraction of the secret cryptographic keys:

- O.CONFIDENTIAL_KEYS ensures that all cryptographic keys are protected against disclosure;
- O.CORRECT_OPERATION ensures the integrity of the execution and prevents unauthorized data disclosure.

T.ABUSE_OPERATION The following objective ensures protection against abuse of functionality:

- O.CORRECT_OPERATION ensures the integrity of the execution and prevents unauthorized modifications and data disclosure.

T.FORCE_UNSAFE_STATE The following objective ensures protection against forced unsafe states:

- O.CORRECT_OPERATION ensures the integrity of the execution and the preservation of a secure state upon failure.

5.3.3 Organizational Security Policies

The rationale of the coverage of the OSP by the security objectives is the following:

OSP.SECRETS The objective OE.SECRETS directly covers this OSP.

OSP.DIVERSIFICATION The objective OE.DIVERSIFICATION directly covers this OSP.

5.3.4 Assumptions

The rationale of the coverage of the assumptions by the security objectives for the TOE environment is the following:

A.PROTECTION_AFTER_DELIVERY The objective OE.PROTECTION_AFTER_DELIVERY directly covers this assumption.

6 EXTENDED REQUIREMENTS

6.1 FCS_RNG – Generation of random numbers

To define the IT security functional requirements of the TOE, an additional family (FCS_RNG) of the Class FCS (cryptographic support) is defined here. This family describes the functional requirements for random number generation used for cryptographic purposes.

FCS_RNG Generation of random numbers

Family behaviour

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

Component levelling

FCS_RNG Generation of random numbers	1

FCS_RNG.1 requires that random numbers meet a defined quality metric.

Management: FCS_RNG.1

No management activities are foreseen.

Audit: FCS_RNG.1

No actions are defined to be auditable.

FCS_RNG.1 Random number generation

Hierarchical to: No other components.

Dependencies: No dependencies.

FCS_RNG.1.1 The TSF shall provide a *[selection: physical, non-physical true, deterministic, hybrid physical, hybrid deterministic]* random number generator that implements: *[assignment: list of security capabilities]*.

FCS_RNG.1.2 The TSF shall provide *[selection: bits, octets of bits, numbers [assignment: format of numbers]]* that meet *[assignment: a defined quality metric]*.

6.2 AVA_SPECIFIC – Vulnerability analysis of Calypso Basic products

The Calypso Basic products vulnerability analysis is an assessment to determine whether potential vulnerabilities identified in the portable objects could allow attackers to violate the SFRs and thus to perform unauthorized access or modification to data or functionality.

The potential vulnerabilities may be identified either during the evaluation of the development, manufacturing, or assembly environments, during the evaluation of the Calypso Basic product specifications, guidance and available implementation representation or by other methods.

The family 'Vulnerability analysis of Calypso Basic Products (AVA_SPECIFIC)' defines requirements for evaluator independent vulnerability search and penetration testing of Calypso Basic products. Penetration testing is performed by the evaluator assuming Enhanced-basic attack potential.

Component levelling

AVA_SPECIFIC Vulnerability analysis of Calypso Basic products	1

AVA_SPECIFIC.1 Vulnerability analysis of Calypso Basic products

Dependencies: ADV_ARC.1 Security architecture description

ADV_FSP.2 Security-enforcing functional specification

ADV_TDS.1 Basic design

AGD_OPE.1 Operational user guidance

AGD_PRE.1 Preparative procedures

Objectives

A vulnerability analysis is performed by the evaluator to ascertain the presence of potential vulnerabilities.

The evaluator performs penetration testing on the Calypso Basic product to confirm that the potential vulnerabilities cannot be exploited in the operational environment of the Calypso Basic product. Penetration testing is performed by the evaluator assuming Enhanced-basic attack potential.

Note: Underlined text highlights the differences against AVA_VAN.2.

Developer action elements:

AVA_SPECIFIC.1.1D The developer shall provide the TOE for testing.

Content and presentation elements:

AVA_SPECIFIC.1.1C The TOE shall be suitable for testing.

Evaluator action elements:

AVA_SPECIFIC.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_SPECIFIC.1.2E The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the TOE.

AVA_SPECIFIC.1.3E The evaluator shall perform an independent vulnerability analysis of the TOE using the guidance documentation, functional specification, TOE design, security architecture description and TSF implementation representation to identify potential vulnerabilities in the TOE.

AVA_SPECIFIC.1.4E The evaluator shall conduct penetration testing, based on the identified potential vulnerabilities, to determine that the TOE is resistant to attacks performed by an attacker possessing Enhanced-basic attack potential.

7 SECURITY REQUIREMENTS

7.1 Security Functional Requirements

This Protection Profile uses the following standard security functional components:

- FCS_CKM.1 Cryptographic key generation;
- FCS_CKM.4 Cryptographic key destruction;
- FCS_COP.1 Cryptographic operations;
- FDP_ACC.1 Subset access control;
- FDP_ACF.1 Security attribute based access control;

- FDP_RIP.1 Subset residual information protection;
- FDP_ROL.1 Basic Rollback;
- FMT_MSA.1 Management of security attributes;
- FMT_MSA.2 Secure security attributes;
- FMT_MSA.3 Static attribute initialization;
- FMT_MTD.1 Management of TSF data;
- FMT_MTD.2 Management of limits on TSF data;
- FMT_MTD.3 Secure TSF data;
- FMT_SMF.1 Specification of management functions;
- FMT_SMR.1 Security roles;
- FPT_FLS.1 Failure with preservation of secure state;
- FTP_ITC.1 Inter-TSF trusted channel.

Moreover, the following extended security functional components, defined in the previous sub-section, are used:

- FCS_RNG.1 Random numbers generation.

Some of the SFRs defined in this PP carry open assignment operations which must be filled in by the ST author. Some of these operations allow the ST author to cover implementation-dependent features that are beyond the Calypso Basic specification [C-BASIC], for which an empty assignment is also acceptable. This is indicated in Application Notes.

7.1.1 FCS_CKM.1 Cryptographic key generation

FCS_CKM.1.1 The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm *[assignment: cryptographic key generation algorithm]* and specified cryptographic key sizes of *[assignment: cryptographic key sizes]* that meet the following: *[assignment: list of standards]*.

Application Note: This SFR applies to the Calypso Basic session keys. The Calypso Basic specification [C-BASIC] ensures the uniqueness and unpredictability of the session keys by using a challenge that includes the TC and a random number.

7.1.2 FCS_CKM.4 Cryptographic key destruction

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method *[assignment: cryptographic key destruction method applied to the Calypso Basic session keys]* that meet the following: *[assignment: list of standards]*.

Application Note: The static application keys cannot be destroyed or replaced. The destruction of Calypso Basic session keys does not require to comply with a specific standard. The present SFR allows the ST author to specify the session keys destruction method that is used, if any (an empty assignment is also acceptable).

7.1.3 FCS_COP.1 Cryptographic operation

FCS_COP.1.1 The TSF shall perform **Session MAC authentication** in accordance with a specified cryptographic algorithm **MAC Algorithm 1 with Padding method 2** and cryptographic key sizes of **112-bit (TDES keys option 2)** that meet the following: **ISO/IEC 9797-1 and ISO/IEC 18033-3.**

Application Note: MAC Algorithm 1 with Padding method 2 is covered in ISO/IEC 9797-1, and TDES keying option 2 is covered in ISO/IEC 18033-3.

Application Note: The Calypso Basic specification [C-BASIC] ensures the uniqueness and the unpredictability of the MAC by including the TC and a random number. The cryptographic keys used for the Session MAC authentication are those generated with FCS_CKM.1.

7.1.4 FCS_RNG.1 Random Number Generation

FCS_RNG.1.1 The TSF shall provide a *[selection: physical, non-physical true, deterministic, hybrid physical, hybrid deterministic]* random number generator that implements: *[assignment: list of security capabilities]*.

FCS_RNG.1.2 The TSF shall provide *[selection: bits, octets of bits, numbers [assignment: format of numbers]]* that meet *[assignment: a defined quality metric].*

Application Note: The definition of the quality metric should meet recognized standards, see for instance [SOGIS-ACM].

7.1.5 FDP_ACC.1 Subset access control

FDP_ACC.1.1 The TSF shall enforce the **Calypso Basic access control SFP** on:

- Subjects: **Calypso Basic Application** *[assignment: other subjects of the Calypso Basic product]*
- Objects: **Files** *[assignment: other persistent or transient objects of the Calypso Basic product]*
- Operations: **read(), modify(), invalidate()** *[assignment: other Calypso Basic product operations].*

Application Note: The correspondence with Calypso Basic commands is the following:

- read() stands for read_record(): reads a single record from the indicated file;
- modify() stands for
    - write_record(): writes over the data of the indicated record of a file;
    - append_record(): adds a record to a Cyclic file; this record becomes the first record of the file; the last record is removed;
    - update_record(): replaces the data of the indicated record with the new data provided;
    - increment_value(): increments the value of a Counter file;
    - decrement_value(): decrements the value of a Counter file.
- invalidate() stands for the command with the same name, which invalidates the DF; when the DF is invalidated, any command modifying the file system data is rejected.

Application Note: There are other commands in the Calypso Basic product that are not concerned or affected by the access control to the file's contents:

- select_application(): selects an application in the Calypso Basic product making the DF of the application the current DF and the current file and returns information about this application; aborts any secure session currently opened; if the command fails, the current file and current DF remain unchanged;
- select_file(): gets the current DF or sets the current file pointer to a specific EF; returns the type and parameters of the file;
- open_secure_session(key.id, file_id): opens a secure session with the indicated key key.id; when a file identifier file_id is indicated, it becomes the current file; otherwise, the currently selected file is used;
- close_secure_session(): closes the currently open secure session; if the Session MAC high part is correct, the modifications are committed, otherwise, the modifications are rollbacked;
- get_data(): returns the requested data.

Application Note: The usage scenarios can be characterized by a sequence of operations denoted by ';' in the following examples:

- Ticket-loading: select_application(); open_secure_session(); if (transaction_counter > 0) then read_record() and (update_record() and/or write_record() and/or append_record() and/or increment_value()); close_secure_session()
- Ticket-control: select_application(); open_secure_session(); read_record(); close_secure_session().
- Ticket-debiting: select_application(); open_secure_session();if (transaction_counter > 0) then (read_record(); decrement_value(); append_record()); close_secure_session().

Application Note: The assignments allow the ST author to cover implementation-dependent subjects, objects and operations which are beyond the Calypso Basic specification [C-BASIC]. Empty assignments are acceptable.

7.1.6 FDP_ACF.1 Security attribute based access control – v0.8

FDP_ACF.1.1 The TSF shall enforce the **Calypso Basic access control SFP** to objects based on the following:

- **Security attributes of the Calypso Basic Application:**
  - **session_status (closed/1:Issuer/2:Load/3:Debit)**
  - **DF_status (valid/invalid)**
  - **access_rights: file x operation → {Always, Session_1, Session_2, Session_3, Never, undefined}**
  - *[assignment: list of additional security attributes]*

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

- **DF_status = invalid and (op = invalidate or op=read), or**
- **DF_status = valid and access_rights(f,op) = Always, or**
- **DF_status = valid and (access_rights(f,op) = Session_i and session_status = j and j <= i)**

where op is the operation requested over the file f

- *[assignment: list of additional rules]*.

FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based on the following additional rules*: [assignment: list of additional rules].*

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **at least one of the conditions stated in FDP_ACF.1.2 is not satisfied, that is:**

- **DF_status = invalid and op = modify, or**
- **access_rights(f,op) = Never or undefined, or**
- **access_rights(f,op) = Session_i and session_status = j and j > i**

where op is the operation requested over the file f

- *[assignment: list of additional rules].*

Application Note: The value 'undefined' is introduced to fully cover the domain (file_type x operation).

Application Note: The assignments allow the ST author to cover implementation-dependent security attributes and rules which are beyond the Calypso Basic specification [C-BASIC]. Empty assignments are acceptable.

7.1.7 FDP_RIP.1 Residual information protection

FDP_RIP.1.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the **deallocation of the resource** from the following objects: **session keys, Session MAC, random numbers** *[assignment: list of objects].*

Application Note: The assignment allows the ST author to cover implementation-dependent objects which are beyond the Calypso Basic specification [C-BASIC]. An empty assignment is acceptable.

7.1.8 FDP_ROL.1 Basic rollback

FDP_ROL.1.1 The TSF shall enforce the **Calypso Basic access control SFP** to permit the rollback of the **modify() and invalidate() operations** on the **Calypso Basic files and DF_status, respectively, without modifying the Transaction Counter** *[assignment : list of objects].*

FDP_ROL.1.2 The TSF shall permit operations to be rollbacked within **the limits of the set of operations performed in the failed or interrupted secure session.**

Application Note: Recall that modify() stands for write_record(), append_record(), update_record(), increment_value() and decrement_value().

Application Note: When the Calypso Basic Application cannot authenticate the Terminal, i.e. when the Session MAC high is incorrect, the secure session fails. It can also be interrupted/cancelled by the Calypso Basic Application for other reasons.

Application Note: The assignment allows the ST author to cover implementation-dependent objects which are beyond the Calypso Basic specification [C-BASIC]. An empty assignment is acceptable.

7.1.9 FMT_MSA.1 Management of security attributes

FMT_MSA.1.1 The TSF shall enforce the Calypso Basic access control SFP to restrict the ability to modify the security attributes session_status, DF_status and access_rights to **the following roles:**

- **Calypso Basic Application in a secure session role for session_status and DF_status**
- **No role for access_rights.**

7.1.10 FMT_MSA.3 Static attribute initialization

FMT_MSA.3.1 The TSF shall enforce the **Calypso Basic access control SFP** to provide **restrictive** default values for security attributes used to enforce the SFP.

FMT_MSA.3.2 The TSF shall allow **no role** to specify alternative initial values to override the default values when an object or information is created.

Application Note: Default values are given upon selection of Calypso Basic application:

- access_rights = fixed in the file structure
- DF_status = last status
- session_status = closed.

7.1.11 FMT_MTD.1 Management of TSF data

FMT_MTD.1.1 The TSF shall restrict the ability to **modify** the **list of the following TSF data** to **the following authorized identified roles:**

- **modify the Calypso Serial Number to no role after Phase 2**
- **modify the Calypso Basic file structure to no role after Phase 2**
- **modify the file access conditions to no role after Phase 2**
- **modify the static application keys or key index to no role after Phase 3**
- **modify the Ratification Status to 'not-ratified' to the Calypso Basic Application in a secure session**
- **modify (decrement) the Transaction Counter to the Calypso Basic Application in a secure session**
- **modify (once) the DF status to the Calypso Basic Application**
- **export static or session application keys to no role.**

Application Note: A secure session request consists in the Calypso Basic Application successfully receiving an open_secure_session() command.

7.1.12 FMT_MTD.2 Management of limits on TSF data

FMT_MTD.2.1 The TSF shall restrict the specification of the limits for **Transaction Counter** to **no role**.

FMT_MTD.2.2 The TSF shall take the following actions, if the TSF data are at, or exceed the indicated limits:

- **If Transaction Counter is '0' then the TSF shall not accept any request to open a secure session**
- **If DF status is 'invalid' then the TSF shall not allow any file modification.**

7.1.13 FMT_MTD.3 Secure TSF data

FMT_MTD.3.1 The TSF shall ensure that only secure values are accepted for

- **Key index**
- **Transaction Counter lower and upper bounds**
- **Transaction Counter**
- **DF status**
- **Keys used for MAC calculation.**

Application Note: In the context of the Calypso Basic Application, a secure value is a value that is managed internally and exclusively by the TSF, i.e. no imported data is accepted. The Calypso Basic specification [C-BASIC] provides the following values restrictions:

- Key Index '1', '2' or '3'
- Transaction Counter upper bound equal to '1000'
- Transaction Counter higher than '0'
- Transaction Counter strictly decreasing
- DF status 'valid' or 'invalid'
- DF status 'invalid' irreversible
- Keys used for MAC calculation are generated as specified in FCS_CKM.1.

7.1.14 FMT_SMR.1 Security roles

FMT_SMR.1.1 The TSF shall maintain the roles

- **Calypso Basic Application in a secure session**
- **Calypso Basic Application out of a secure session.**

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

Application Note: here 'users' stands for the Terminal. The association is initiated upon reception of an open_secure_session command provided the Transaction Counter is higher than 0 and the DF status is valid.

7.1.15 FPT_FLS.1 Failure with preservation of secure state

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

- **Random number generation failure (FCS_RNG.1)**
- **Cryptographic key generation failure (FCS_CKM.1)**
- **Cryptographic operation failure (FCS_COP.1)**
- **File access control failure (FDP_ACC.1, FDP_ACF.1)**
- **Transaction Counter reached the lower limit (FMT_MTD.2)**
- **Secure session failure (FTP_ITC.1)**
- **Rollback failure (FDP_ROL.1)**
- **Deallocation failure (FDP_RIP.1)**

- **TSF data protection failure (FMT_MSA.1, FMT_MTD.1, FMT_MTD.3)**
- **Unexpected or unknown commands**
- **Unexpected termination**
- *[assignment: list of types of failures in the TSF].*

Application Note: A secure state is a state in which the assets' security properties (confidentiality, integrity, etc.) are enforced.

Application Note: The following are examples of failures

- Related to file access control: Failed or aborted modification commands because:
  - DF was set to 'invalid'
  - Access rights were not correctly enforced
- Related to the secure session:
  - Incorrect Session MAC
  - Limit of the allowed number of modifications inside a secure session exceeded
  - Unsuccessful close_secure_session() command
- Related to termination:
  - Card tearing
  - Calypso Basic application selection
  - Calypso Basic application termination.

Application Note: The assignment allows the ST author to cover implementation-dependent types of failures which are beyond the Calypso Basic specification [C-BASIC]. An empty assignment is acceptable.

7.1.16 FTP_ITC.1 Inter-TSF trusted channel

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

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

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for **any modification, i.e. (re)loading and debit/validation transactions and application invalidation**.

Application Note: The trusted IT product with which the TSF communicates using the provided communication channel is a Terminal. The Terminal requests the trusted channel by sending an open_secure_session() command. The trusted channel is used to communicate the commands of the secure session.

Application Note: The Calypso Basic specification [C-BASIC] enforces the protection of the integrity of the channel data but does not provide or allow any confidentiality protection. To avoid any misunderstanding, the protection from disclosure is not considered for Calypso Basic products.

7.2 Security Assurance Requirements

The set of SARs applicable to this Protection Profile consists of EAL2 augmented with ALC_DVS.1 and AVA_SPECIFIC.1.

Assurance Class	EAL 2+ Assurance components	
ADV: Development	ADV_ARC.1 Security architecture description Refinement for ADV_ARC.1.2D: Application keys shall not be stored in ROM All debug interfaces are deactivated before delivery All testing interfaces are deactivated before delivery	
	ADV_FSP.2 Security-enforcing functional specification	
	ADV_TDS.1 Basic design	
AGD: Guidance documents	AGD_OPE.1 Operational user guidance	
	AGD_PRE.1 Preparative procedures	
ALC: Life-cycle support	ALC_CMC.2 Use of a CM system	
	ALC_CMS.2 Parts of the TOE CM coverage	
	ALC_DEL.1 Delivery procedures	
	ALC_DVS. 1 Identification of security measures	+
ASE: Security Target evaluation	ASE_CCL.1 Conformance claims	
	ASE_ECD.1 Extended components definition	
	ASE_INT.1 ST introduction	
	ASE_OBJ.2 Security objectives	
	ASE_REQ.2 Derived security requirements	
	ASE_SPD.1 Security problem definition	
	ASE_TSS.1 TOE summary specification	
ATE: Tests	ATE_COV.1 Evidence of coverage	
	ATE_FUN.1 Functional testing	
	ATE_IND.2 Independent testing – sample	
AVA: Vulnerability analysis	AVA_VAN.2 Vulnerability analysis	
	AVA_SPECIFIC Vulnerability analysis of Calypso Basic products Refinement for AVA_SPECIFIC.1.4E: The definition of Enhanced-basic attack potential is given in [JIL-AAPS].	+

7.3 Security Requirements Rationale

7.3.1 Security Objectives for the TOE

The following table provides an overview for the relationship between the security objectives for the TOE and the SFRs.

SFRs \ Objectives	O.SECURE_SESSION	O.MONOTONIC_TC	O.RNG	O.SESSION_KEYS	O.SESSION_AUTHENTICATION	O.FILE_ACCESS_CONTROL	O.FS_STATUS	O.NON-MODIFIABLE_DATA	O.CONFIDENTIAL_KEYS	O.CORRECT_OPERATION
FCS_CKM.1	x			x						
FCS_CKM.4									x	
FCS_COP.1	x				x					
FCS_RNG.1	x		x							
FDP_ACC.1						x	x			
FDP_ACF.1						x	x			
FDP_RIP.1									x	
FDP_ROL.1	x									
FMT_MSA.1	x					x	x			
FMT_MSA.3						x				
FMT_MTD.1	x	x					x	x	x	
FMT_MTD.2	x	x					x			
FMT_MTD.3	x	x					x			
FMT_SMR.1	x	x				x	x			
FPT_FLS.1										x
FPT_ITC.1	x									

The rationale of the coverage of the security objectives for the TOE by the SFRs is the following:

O.SECURE_SESSION The following requirements contribute to fulfil the objective:

- FCS_CKM.1 states the session key generation conditions needed to be fulfilled for a secure session;
- FCS_COP.1 requires performing Session MAC authentication;
- FCS_RNG.1 states the conditions for the random numbers used for the secure session;
- FDP_ROL.1 stipulates the rollback of a writing operation in case of a failed or interrupted secure session;
- FMT_MSA.1 states the policy for reading and modifying the security attributes linked to a secure session;
- FMT_MTD.1 states the roles that can decrease the Transaction Counter and set the Ratification Status to 'not-ratified' upon a request for opening a secure session;
- FMT_MTD.2 ensures that the Transaction Counter and the DF status are securely managed;
- FMT_MTD.3 ensures that only secure values are accepted for the key index, Transaction Counter and DF_status;
- FMT_SMR.1 specifies the security role to be maintained in and out of a secure session;
- FTP_ITC.1 enforces mutual authentication between the TSF and the terminal.

O.MONOTONIC_TC The following requirements contribute to fulfil the objective:

- FMT_MTD.1 states the roles that can decrease the Transaction Counter;
- FMT_MTD.2 ensures that the Transaction Counter is securely managed;
- FMT_MTD.3 states the secure values that the Transaction Counter may take;
- FMT_SMR.1 states the security role to be maintained in and out of a secure session.

O.RNG The following requirement fulfills the objective:

- FCS_RNG.1 states the conditions for the random number generation used for secure sessions.

O.SESSION_KEYS The following requirement fulfills the objective:

- FCS_CKM.1 states the session key generation conditions needed to be fulfilled for a secure session, including the uniqueness of the session keys.

O.SESSION_AUTHENTICATION The following requirement fulfills the objective:

- FCS_COP.1 requires performing Session MAC authentication and ensuring the uniqueness of the MAC.

O.FILE_ACCESS_CONTROL The following requirements contribute to fulfil the objective:

- FDP_ACC.1 and FDP_ACF.1 state the policy for controlling access to files and operations on them;
- FMT_MSA.1 states the roles that can read the file access conditions upon a request for opening a secure session and FMT_MSA.3 disallows all roles to specify alternative values for the initial file access conditions;
- FMT_SMR.1 specifies the security role to be maintained in and out of a secure session.

O.FS_STATUS The following requirements contribute to fulfil the objective:

- FDP_ACC.1 and FDP_ACF.1 state the policy for accessing the DF status and the invalidate() command;
- FMT_MSA.1 states the policy for setting once the DF status;
- FMT_MTD.1 states the roles that can set once the DF_status to 'invalid' and FMT_MTD.3 ensures that only a secure value is accepted for the DF_status;
- FMT_MTD.2 ensures that the DF status is securely managed;
- FMT_SMR.1 specifies the security role to be maintained in and out of a secure session.

O.NON_MODIFIABLE_DATA The following requirement fulfills the objective:

- FMT_MTD.1 ensures that the Calypso Serial Number and the file structure and file attributes cannot be modified after initialization; it also ensures that the static application keys and key identifiers cannot be modified after pre-personalization.

O.CONFIDENTIAL_KEYS The following requirements contribute to fulfil the objective:

- FCS_CKM.4 ensures that session keys are destroyed as required;
- FDP_RIP.1 ensures that all session keys and random numbers used for generating them are deleted upon deallocation;
- FMT_MTD.1 ensures that the static or session application keys cannot be exported.

O.CORRECT_OPERATION The following requirement fulfills the objective:

- FPT_FLS.1 ensures the preservation of a secure state when a failure occurs.

7.3.2 SFR Dependencies

The following table provides an overview of the SFR dependencies.

SFRs in this PP	CC dependencies	Satisfied dependencies	Unsatisfied dependencies
FCS_CKM.1	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4	FCS_COP.1 FCS_CKM.4	
FCS_CKM.4	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	FCS_CKM.1	
FCS_COP.1	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1 FCS_CKM.4	
FCS_RNG.1	No dependencies		
FDP_ACC.1	FDP_ACF.1	FDP_ACF.1	
FDP_ACF.1	FDP_ACC.1 FMT_MSA.3	FDP_ACC.1 FMT_MSA.3	
FDP_RIP.1	No dependencies		
FDP_ROL.1	[FDP_ACC.1. or FDP_IFC.1]	FDP_ACC.1	
FMT_MSA.1	[FDP_ACC.1. or FDP_IFC.1] FMT_SMR.1 FMT_SMF.1	FDP_ACC.1 FMT_SMR.1	FMT_SMF.1
FMT_MSA.3	FMT_MSA.1 FMT_SMR.1	FMT_MSA.1 FMT_SMR.1	
FMT_MTD.1	FMT_SMR.1 FMT_SMF.1	FMT_SMR.1	FMT_SMF.1
FMT_MTD.2	FMT_MTD.1 FMT_SMR.1	FMT_MTD.1 FMT_SMR.1	
FMT_MTD.3	FMT_MTD.1	FMT_MTD.1	
FMT_SMR.1	FIA_UID.1		FIA_UID.1
FPT_FLS.1	No dependencies		
FTP_ITC.1	No dependencies		

The rationale for the exclusion of SFR dependencies is the following:

For FMT_MSA.1 and FMT_MTD.1: The dependency on FMT_SMF.1 is discarded since there is no management function associated with these SFRs.

For FMT_SMR.1**:** The dependency on FIA_UID.1 is discarded. The Calypso Basic product does not identify Terminals, but it simply disallows some commands before the open_secure_session command is successfully processed. Notice that a successful open_secure_session command does not mean that the Terminal is genuine, since the authentication is made at the end of the secure session, before the product commits the requested changes.

7.3.3 SARs Dependencies

The following table provides an overview of the SAR dependencies and how they are satisfied. There is no unsatisfied dependency.

SAR in this PP	CC dependencies	Satisfied dependencies
ADV_ARC.1 Security architecture description	ADV_FSP.1 Basic functional specification ADV_TDS.1 Basic design	ADV_FSP.2 ADV_TDS.1
ADV_FSP.2 Security-enforcing functional specification	ADV_TDS.1 Basic design	ADV_TDS.1
ADV_TDS.1 Basic design	ADV_FSP.2 Security-enforcing functional specification	ADV_FSP.2
AGD_OPE.1 Operational user guidance	ADV_FSP.1 Basic functional specification	ADV_FSP.2
AGD_PRE.1 Preparative procedures	No dependencies	
ALC_CMC.2 Use of a CM system	ALC_CMS.1 TOE CM coverage	ALC_CMS.2
ALC_CMS.2 Parts of the TOE CM coverage	No dependencies	
ALC_DEL.1 Delivery procedures	No dependencies	
ALC_DVS. 1 Identification of security measures	No dependencies	
ASE_CCL.1 Conformance claims	ASE_INT.1 ST introduction ASE_ECD.1 Extended components definition ASE_REQ.1 Stated security requirements	ASE_INT.1 ASE_ECD.1 ASE_REQ.2
ASE_ECD.1 Extended components definition	No dependencies	
ASE_INT.1 ST introduction	No dependencies	
ASE_OBJ.2 Security objectives	ASE_SPD.1 Security problem definition	ASE_SPD.1
ASE_REQ.2 Derived security requirements	ASE_OBJ.2 security objectives ASE_ECD.1 Extended components definition	ASE_OBJ.2 ASE_ECD.1
ASE_SPD.1 Security problem definition	No dependencies	
ASE_TSS.1 TOE summary specification	ASE_INT.1 ST introduction ASE_REQ.1 Stated security requirements ADV_FSP.1 Basic functional specification	ASE_INT.1 ASE_REQ.2 ADV_FSP.2
ATE_COV.1 Evidence of coverage	ADV_FSP.2 Security-enforcing functional specification ATE_FUN.1 Functional testing	ADV_FSP.2 ATE_FUN.1
ATE_FUN.1 Functional testing	ATE_COV.1 Evidence of coverage	ATE_COV.1
ATE_IND.2 Independent testing – sample	ADV_FSP.2 Security-enforcing functional specification AGD_OPE.1 Operational user guidance AGD_PRE.1 Preparative procedures ATE_COV.1 Evidence of coverage ATE_FUN.1 Functional testing	ADV_FSP.2 AGD_OPE.1 AGD_PRE.1 ATE_COV.1 ATE_FUN.1
AVA_VAN.2 Vulnerability analysis	ADV_ARC.1 Security architecture description ADV_FSP.2 Security-enforcing functional specification ADV_TDS.1 Basic design AGD_OPE.1 Operational user guidance AGD_PRE.1 Preparative procedures	ADV_ARC.1 ADV_FSP.2 ADV_TDS.1 AGD_OPE.1 AGD_PRE.1
AVA_SPECIFIC.1 Vulnerability analysis of Calypso Basic products	ADV_ARC.1 Security architecture description ADV_FSP.2 Security-enforcing functional specification ADV_TDS.1 Basic design AGD_OPE.1 Operational user guidance AGD_PRE.1 Preparative procedures	ADV_ARC.1 ADV_FSP.2 ADV_TDS.1 AGD_OPE.1 AGD_PRE.1

7.3.4 Rationale for the Security Assurance Requirements

The assurance level defined in this Protection Profile consists of the predefined assurance package EAL2 augmented with ALC_DVS.1 and AVA_SPECIFIC.1, which ensures the protection of the assets in the development/manufacturing environment and allows to reach the Enhanced-basic attack potential as defined in [JIL-AAPS].

This EAL2+ package is compatible with industry best practices and allows to gain sufficient assurance for Calypso Basic products. The developer provides the specification and design, the test plans and results, the guidance and delivery procedure and the evidence related to configuration management system as required by EAL2. Moreover, the developer provides evidence of the development/manufacturing site security to cover skimming attacks for instance, and the source code of the TSF as required in AVA_SPECIFIC.1.

By comparison, the standard component AVA_VAN.2 included in EAL2 ensures resistance to Basic attack potential, which is considered insufficient for Calypso Basic Products as relevant attack scenarios would be out of reach. Indeed, Calypso Basic products are expected to resist to Enhanced-basic attack potential as defined in [JIL-AAPS] and the source code must be available to perform the vulnerability analysis. Note that the use of the standard AVA_VAN.3 with CC dependencies is not compatible with the trade-off between developer's effort and assurance expected by CNA for Calypso Basic products. This is the reason for defining AVA_SPECIFIC.1.

The components AVA_VAN.2 and AVA_SPECIFIC.1 are both included in the EAL2+ package.