# EURO-MILS

## Secure European Virtualisation for Trustworthy Applications in Critical Domains

# Common Criteria Protection Profile

| | |
|---|---|
| **Project number** | 318353 |
| **Project acronym** | EURO-MILS |
| **Project title** | EURO-MILS:<br>Secure European Virtualisation for Trustworthy Applications in Critical Domains |
| **Start date of the project** | 1st October, 2012 |
| **Duration** | 36 months |
| **Programme** | FP7/2007-2013 |
| **Project website** | www.euromils.eu |

Common Criteria Protection Profile

registration ID

**Editors/Authors:** Igor Furgel, Viola Saftig; T-Systems International GmbH (TSYS)

Further information on the EURO-MILS Project: http://www.euromils.eu

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 318353.

# MILS Protection Profile

# Whitepaper 2015

## Executive Summary

This Protection Profile ‘Multiple Independent Levels of Security: Operating System (MILS PP: Operating System)’ is issued by the EURO-MILS Consortium.

This PP addresses only Operating System as part of a MILS final integrated system. In the future there may be also other PPs regarding MILS architecture, like hardware platform or the entire integrated system.

The TOE, as addressed in the current PP, does not include any hardware. If it is desired to certify a TOE also comprising hardware components, the related ST will include these hardware components as part of the TOE. If appropriate, the re-assignment operation may be applied:

”The ST may specify that certain objectives for the operational environment in the PP are security objectives for the TOE in the ST. […] If a security objective is re-assigned to the TOE the security objectives rationale has to make clear which assumption or part of the assumption may not be necessary any more“ ([1], chapter 9.3).

The document has been prepared as a Protection Profile (PP) following the rules and formats of Common Criteria version 3.1 [1], [2], [3], Revision 4.

# Table of Content

# Chapter 1 PP Introduction

1 This section provides document management and overview information required to register the protection profile and to enable a potential user of the PP to determine, whether the PP is of interest.

## 1.1 PP reference

| | | |
|---|---|---|
| 2 | Title: | Protection Profile<br>'Multiple Independent Levels of Security: Operating System<br>(MILS PP: Operating System)' |
| | Sponsor: | EURO-MILS Consortium |
| | Editor(s): | Dr. Igor Furgel, Viola Saftig<br>T-Systems International GmbH (TSYS) |
| | CC Version: | 3.1 (Revision 4) |
| | Assurance Level: | Minimum assurance level for this PP is EAL5 augmented. |
| | General Status: | released |
| | Version Number: | 1.02 as of 12th March 2015 |
| | Registration: | registration ID |
| | Keywords: | Operating system, Separation kernel, MILS (Multiple Independent Levels of Security), Virtualization, Hypervisor |

## 1.2 TOE Overview

### *1.2.1 TOE definition and operational usage*

3 The Target of Evaluation (TOE) addressed by the current protection profile is a special kind of operating system, that allows to effectively separate different applications running on the same platform from each other.

4 The TOE can host user applications that can also be operating systems. User applications can even be malicious, and even in that case the TOE ensures that malicious user applications are neither harming the TOE nor other applications in other partitions. The TOE will be installed and run on a hardware platform (e.g. embedded systems).

5 The TOE is intended to be used as a component (the separation kernel) in MILS systems. MILS (Multiple Independent Levels of Security) systems are explained in [9], [10] and [11].

6 The TOE controls usage of memory, devices, processors, and communication channels to ensure *complete separation* of user applications and to prevent unexpected interference between user applications. The TOE enforces restrictions on the communication between the separated user applications as specified by the configuration data.

### 1.2.2 TOE type

7 The TOE is a special kind of operating system providing a separation kernel with real-time support.

8 The typical *life cycle* phases for this TOE type are development (source code development), manufacturing (compilation to binary), system integration (by the system integrator), installation (by the system operator), and finally, operational use (by the system operator). Operational use of the TOE is explicitly in the focus of this PP. A security evaluation/certification according to the assurance package chosen in this PP (see the statement“This PP does not claim conformance to any protection profile” in Section 2.1 unterhalb) involves all these life cycle phases.

### 1.2.3 Non-TOE hardware/software/firmware

9 The TOE may run on various hardware platforms. The TOE, as addressed in the current PP, does not include any hardware. If it is desired to certify a TOE also comprising hardware components, the related ST will include these hardware components as part of the TOE. The minimum requirements on the CPU of the hardware platform are a memory management unit (MMU) and support for different CPU privilege modes.

10 **Explanatory Note 1:** Obligations on hardware usage are given in Section 3.3, organizational security policies P.SYSTEM_INTEGRATOR and P.SYSTEM_OPERATOR.

11 **Explanatory Note 2:** If it is desired to certify a TOE also comprising hardware components, the related ST will include these hardware components as part of the TOE. See also “Foreword”.

## 1.3 TOE Description

### 1.3.1 *TOE Architecture*

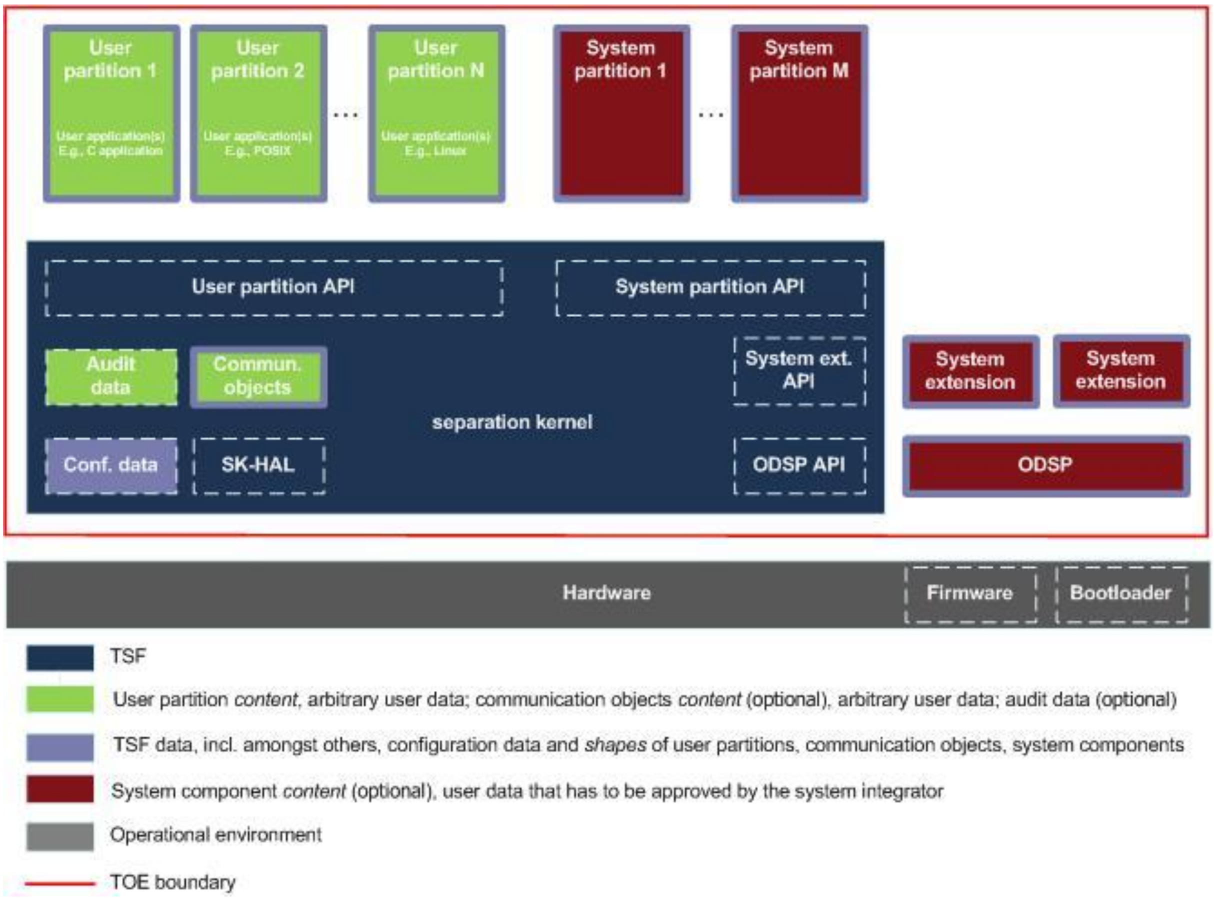
User partition 1
User application(s) E.g., C application
User partition 2
User application(s) E.g., POSIX
...
User partition N
User application(s) E.g., Linux
System partition 1
...
System partition M
User partition API
System partition API
Audit data
Commun. objects
System ext. API
separation kernel
Conf. data
SK-HAL
ODSP API
System extension
System extension
ODSP
Hardware
Firmware
Bootloader
TSF
User partition *content*, arbitrary user data; communication objects *content* (optional), arbitrary user data; audit data (optional)
TSF data, incl. amongst others, configuration data and *shapes* of user partitions, communication objects, system components
System component *content* (optional), user data that has to be approved by the system integrator
Operational environment
TOE boundary

Figure 1: TOE and TOE Operational Environment During Operational Use

12 Figure 1, especially the difference between green and red components, will be explained in detail in the next sections (Section 1.3.2 and 1.3.3).

### 1.3.2 *TOE*

13 The TOE, delineated within the red line in Figure 1 consists of a separation kernel (TSF), TSF data and user data. The separation kernel and TSF data represent the TOE operating system.

#### 1.3.2.1 TOE Operating System

14 The separation kernel provides the TSF and operates the TOE, by implementing mechanisms to assign resources to partitions, providing the execution environments for applications, and implementing communication between partitions as defined by the configuration data.

15 The separation kernel provides Application Programming Interfaces (APIs) to user partitions and system partitions as well as APIs to system extensions and on-board device support package (ODSP).

16 A Separation Kernel Hardware Abstraction Layer (SK-HAL) provides specific low-level functionality for each supported CPU architecture.. In operational use, the TOE always contains only one SK-HAL.

17 TSF data consists of

- Configuration data: Data used by the TSF to enforce the System Security Policy (SSP, Section 1.3.4.2), depicted as a bright blue box in Figure 1.
- Shape data: A shape is TSF data that contains an entity's identity, the entity's resource usage data, a set of security attributes according to the SSP assigned to the entity, and links the content assigned to an entity to the resources assigned to the entity (Section 3.1.1.2). Shapes are depicted as bright blue frames in Figure 1.

#### 1.3.2.2 Partition

18 A partition is a logical unit maintained by the separation kernel and configured by the configuration data. A partition contains user data. For each partition, the separation kernel provides resources. Resources of a partition comprise physical memory space, I/O memory space, a description of the set of CPUs the partition's applications can run on, allocated CPU time for each CPU, and interrupts.

19 The TOE supports two different kinds of partitions: user and system partitions. User partitions, depicted as green content surrounded by bright blue shapes in Figure 1, are defined in Section 1.3.2.2.1. System partitions, depicted as red content surrounded by bright blue shapes in Figure 1, are defined in Section 1.3.2.2.2.

20 Partitions can communicate with each other under the supervision of the TOEs separation kernel. This communication occurs via communication objects. A communication object is an object exposed to one or multiple partitions with access rights as defined in the configuration data.

##### 1.3.2.2.1User Partition

21 User partition: A *user partition* contains user applications and/or data being executed and/or stored in a user partition. User applications can be arbitrary and even malicious. User applications use the user partition API of the separation kernel. The content of a user partition does not have to be approved by the system integrator. The content of a user partition can be exchanged without changing the separation kernel binary image, the content of any other partition or the content of a system component of the TOE, see Section 1.3.4.2.

#### 1.3.2.2.2System Partition

22 System partition: A *system partition* contains applications and/or data supplied and approved by the system integrator. An application in a system partition is a *system application* and uses the system partition API of the separation kernel. The content of a system partition can be exchanged without changing the separation kernel binary image, the content of any other partition or the content of a system component of the TOE.

23 **Explanatory Note 3**: The ability of the TOE to support system partitions is optional and a ST/PP compliant to this PP can choose to have system partitions or not to have system partitions. The author of the related ST/PP shall clearly state it.

### 1.3.2.3 System Component

24 A *system component* is a system partition (Section 1.3.2.2.2 above), system extension (Section 1.3.2.4 below), or an ODSP (Section 1.3.2.5 below). A system component contains user data supplied and approved by the system integrator.

### 1.3.2.4 System Extension

25 System extension: A *system extension* contains a software component (a system application) supplied and approved by the system integrator and coupled with the separation kernel via the system extension API. A system extension can provide specific functionality to applications within partitions only under supervision of the separation kernel. A system extension can be exchanged without changing the separation kernel binary image, the content of any other partition or the content of a system component of the TOE.

26 **Explanatory Note 4**: The ability of the TOE to support system extensions is optional and a ST/PP compliant to this PP can choose to have system extensions or not to have system extensions. The author of the related ST/PP shall clearly state it.

### 1.3.2.5 On-board Device Support Package (ODSP)

27 On-board device support package: An *on-board device support package* is a special purpose HAL and may contain a set of drivers for specific hardware components (a system application). It is supplied and approved by the system integrator. An *on-board device support package* can be exchanged without changing the separation kernel binary image, the content of any other partition or the content of a system component of the TOE. An *on-board device support package* uses the TSF's *on-board device support package* API. In operational use, the TOE always contains only one *on-board device support package*.

### 1.3.2.6 Audit Data

28 *Audit data* is user data consisting of electronic records reflecting events to be audited.

29 **Explanatory Note 5**: The ability of the TOE to support the generation of audit data is optional and a ST/PP compliant to this PP can choose to have the generation of audit data or not. The author of the related ST/PP shall clearly state it.

### 1.3.2.7 Communication object

30 A *communication object* contains user data. See Section 1.3.2.2

31 **Explanatory Note 6**: If a concrete TOE implementation cannot principally use any communication objects, the author of the related ST/PP shall clearly state it. Such a TOE implementation is considered to be compliant to this PP.

### *1.3.3 TOE Operational Environment*

32 The TOE operational environment, outside the red line in Figure 1, consists of:

33 Hardware: *Hardware* is the physical part of the TOE operational environment on which the TOE is executed. Usually, hardware is a board with several components such as CPUs, serial interfaces, network adapters, I/O devices etc. There are Separation Kernel Hardware Abstraction Layer  controlled components (e.g. CPUs, caches) and ODSP controlled components (e.g. serial interfaces, timer).

34 Hardware may also comprise the following hardware-specific software:

- Firmware: *Firmware* is software and data stored in non-volatile memory of the hardware that initializes the hardware after the power on.
- Bootloader: A *bootloader* is software that loads the TOE on the hardware and hands over the full control to the TOE. In particular, a TOE-external check of the TOE may be implemented in the bootloader (e.g. for “secure boot”).

### *1.3.4 TOE Life Cycle*

The generic lifecycle of the TOE comprises of development/manufacturing, System Integration, Installation and Operational Use.

#### 1.3.4.1 Development, Manufacturing

35 At the TOE manufacturer’s site the TSF is developed (source code development), and manufactured (compiled to binary). The TOE manufacturer also produces the TOE User Manuals.

#### 1.3.4.2 System Integration

36 At the system integrator’s site, the TOE is integrated. Figure 2 presents the generic Lifecycle of the TOE. Components used to build the product based on the TOE are provided by different sources: user application developers, system integrators, and the TOE manufacturer.

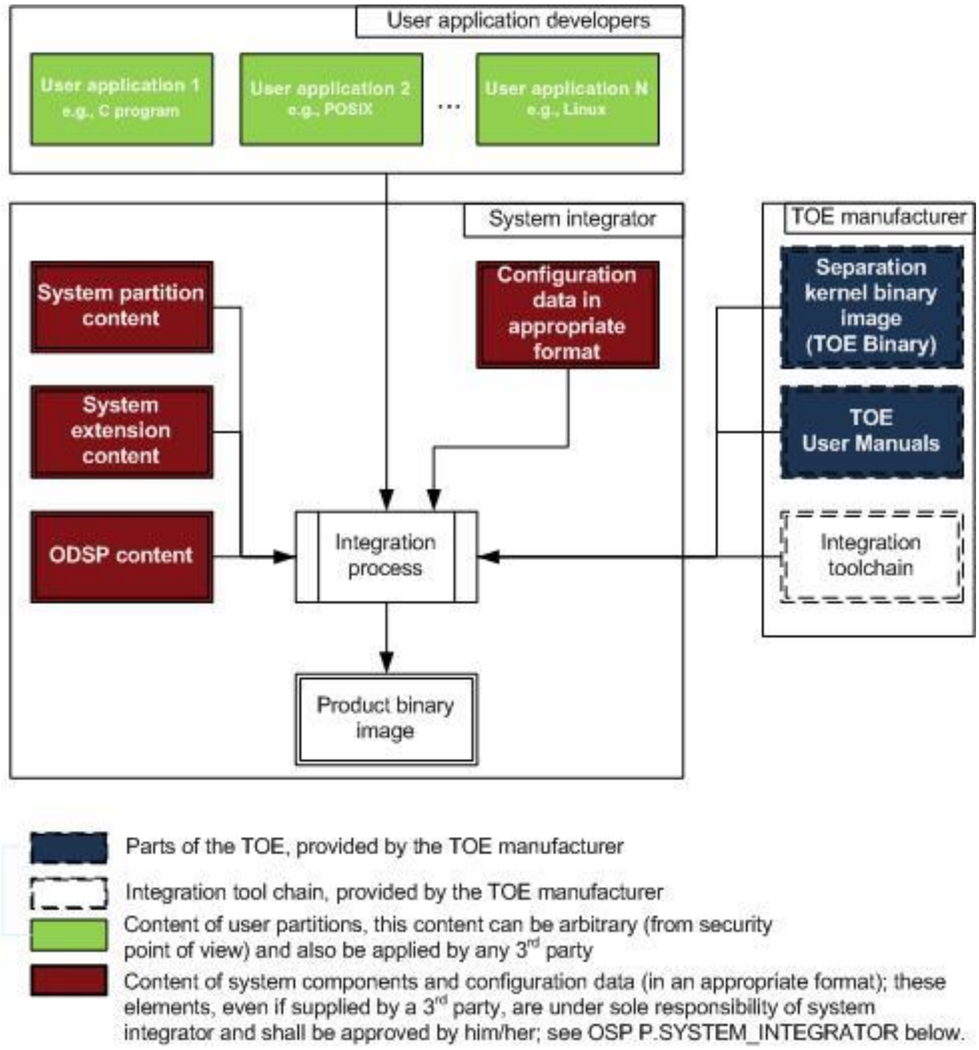
User application developers
User application 1
e.g., C program
User application 2
e.g., POSIX
...
User application N
e.g., Linux
System integrator
System partition content
System extension content
ODSP content
Configuration data in appropriate format
Integration process
Product binary image
TOE manufacturer
Separation kernel binary image (TOE Binary)
TOE User Manuals
Integration toolchain
Parts of the TOE, provided by the TOE manufacturer
Integration tool chain, provided by the TOE manufacturer
Content of user partitions, this content can be arbitrary (from security point of view) and also be applied by any 3rd party
Content of system components and configuration data (in an appropriate format); these elements, even if supplied by a 3rd party, are under sole responsibility of system integrator and shall be approved by him/her; see OSP P.SYSTEM_INTEGRATOR below.

Figure 2: **Generic Lifecycle of the TOE.**

37 The system integration phase of the generic lifecycle can be split into the three steps: selection of the TOE operational environment and system applications and user applications (Step 1), configuration of the TOE (Step 2), and integration (Step 3).

38 The outcome of Step 2 is referred to as configuration data. The *configuration data* defines a set of rules on how the TOE behaves. For example, the configuration data comprises the assignment of resources and communication objects to partitions. The *System Security Policy* (SSP) consists of configuration choices made by a system integrator based on the subset of the configuration data rules evaluated in this PP (for details: see this section, below, in the description of Step 2). The SSP is enforced by the TSF and it cannot be circumvented by malicious user applications.

39 The *combined* outcome of Step 1 and Step 2 is referred to as the *System Integration Policy* (SIP). The SIP comprises user applications and user data that needs to be approved by the system integrator (the content of the ODSP, system partitions, system extensions) and system integration covering hardware choices. See P.SYSTEM_INTEGRATOR for details.

40 **Explanatory Note 7**: The system integrator can derive a *Partitioned Information Flow Policy* (PIFP) from the SSP, applying the following rule: A PIFP information flow from a partition *A* to a partition *B* is allowed if and only if there exists a communication object in the SSP that *A* may modify to and *B* may query.

41 Step 1 Selection

The system integrator selects hardware, and if applicable, firmware and bootloader the TOE runs on.

The system integrator selects the content of components: ODSP, optional system extension(s), optional system partition(s), and user partition(s) to be integrated in the TOE.

The content of any user partition is arbitrary and can be provided by arbitrary application developers.

The content of the ODSP, any system extension, any system partition shall be developed complying with the obligations given in Section 3.3, organizational security policy P.SYSTEM_INTEGRATOR and be approved by the system integrator.

42 Step 2 Configuration

The system integrator configures the product by, for example,

- defining user partitions, setting their content, shapes (see Glossary) and resources,
- defining communication objects, setting their shapes and resources,
- defining system components, setting their content, shapes and resources,
- hardware selection parameters,
- setting TOE attributes, comprising
  - scheduling scheme,
  - policy for memory cache handling on a partition switch to the extent supported by the operational environment’s hardware,
  - scheme for automatic handling of error conditions, defining the meaning of the safe and secure state,
  - configuration of management functions; the audit function is the only one.

The result of this activity is a representation in appropriate format of the configuration data.

The default configuration is that there is no information flow between any partitions. Any information flow between partitions has to be explicitly allowed by the system integrator in the configuration data.

The configuration data uniquely defines the System Security Policy (SSP). The SSP is defining user partitions, setting their shapes and resources, defining communication objects, setting their shapes and resources, defining system components, setting their content, shapes and resources, hardware selection parameters, setting TOE attributes, comprising scheduling scheme, policy for memory cache handling on a partition switch to the extent supported by the operational environment’s hardware, scheme for automatic handling of error conditions, configuration of management functions; the audit function is the only one. An example for a rule defined by the configuration data but not in the SSP is the content of user partitions.

The result of performing Step 2 is that the configuration data has been defined. The result of performing Step 1 and Step 2 is that a SIP has been defined.

43 Step 3 Integration

The system integrator uses the integration tool chain to create a product binary image according to the SIP from the selected components and the representation in appropriate format of the TOE configuration data. The tool chain

- imports, into the user partitions user applications and/or data,
- imports, into system partitions applications and/or data supplied by the system integrator,
- links the content of the on-board device support package and the content of system extensions with the TOE separation kernel binary image, creating the product binary image, including configuration data in a representation readable by the product binary image.

#### 1.3.4.3 Installation

44 The system integrator provides this product binary image to the system operator who, at the system operator's site, installs it on the hardware.

#### 1.3.4.4 Operational Use

45 At the system operator's site, the TOE is operated. At power on the hardware is initialized, then the product binary image is loaded. Immediately after the product binary has been loaded, the on-board device support package, being part of the product binary image, gets invoked. The on-board device support package then starts the TOE separation kernel (TSF), also being part of the product binary image, which initializes itself and starts enforcing the SSP. During operational use, user applications cannot change the product binary image, e.g. no new user or system partitions can be created, no new communication objects can be created, no new user or system applications can be loaded.

### *1.3.5 TOE Physical Boundary*

46 The TOE is a software product. In Figure 1, each component within the red line is within the TOE physical boundary. Each component outside of the red line is outside of the TOE physical boundaries. Thus, no hardware belongs to the TOE. The TOE also includes the TOE User Manuals.

47 **Explanatory Note 8:** If it is desired to certify a TOE also comprising hardware components, the related ST will include these hardware components as part of the TOE. See also "Foreword".

### *1.3.6 TOE Logical Boundary*

48 The TOE provides the following TOE security services, abbreviated as TSS_XXX:

- TSS_SSA: Separation in space of applications hosted in different partitions from each other and from the TOE operating system.

Applications can be hosted in different partitions. Partitions get assigned resources (i.e. space) according to the SSP, which comprise memory ranges and a set of CPUs. The TSF enforces the corresponding part of the SSP by the enforcement of access control on partition content, per-partition provision of physical memory space, I/O memory space, a description of the set of CPUs the partition's applications can run on, and interrupts.

By confining applications into user partitions, the TSF enforces that these applications can affect neither applications in other partitions (user or system applications) nor the TOE operating system itself.

- TSS_STA: Separation in time of applications hosted in different partitions from each other and from the TOE operating system.

  Applications can be hosted in different partitions. Partitions get assigned CPU time (i.e. time windows) according to the SSP. The TSF enforces the corresponding part of the SSP by per-partition allocation of a predefined amount of CPU time for each CPU. Several user and/or system partitions can share the same time window. On a partition switch CPUs will be reused. The TSF enforces that no residual information is in CPU registers or memory caches according to the SSP. The TSF assigns a priority to every subject to allow priority based scheduling within one time window.

- TSS_COM: Provision and management of communication objects.

  Applications hosted in different partitions can get assigned a set of communication objects. A communication object is an object exposed to one or multiple partitions with access rights as defined in the configuration data, thus allowing communication between partitions.

- TSS_MAN: Management of and access to the TSF and TSF data.

  The TSF restricts access to TSF data. Resource usage data is data accounting for the usage of resources. For example, the partition resource usage data accounts for how much memory a partition has already used and how much there is still available. Resource usage data is stored in shapes. The TSF protects the confidentiality and integrity of resources and shapes. The TSF restricts the invokability of the system application API to system applications. Management functions are used for the management of the security behavior of the TSF. The management functions as configured in the SSP can only be invoked by system applications, but can never be invoked by user applications.

- TSS_SPT: TSF self-protection and accuracy of security functionality.

  TSF self-protection and accuracy of functionality supports reaching and keeping a safe and secure state of the TOE. The TSF statically assigns automatic invocations of error handling functions to recover from or respond to error conditions.

- TSS_AUD: Generation and treatment of audit data according to the SSP.

  The TOE separation kernel provides a function for the start-up and shutdown of the audit functions. When the audit function is active the system collects audit data on events to be audited as defined by the SSP, including the date and time of the event, type of event, subject identity (if applicable), and the outcome (success or failure) of the event. Audit data can be queried by user applications and treated by system applications according to the SSP.

# Chapter 2 Conformance Claims

## 2.1 CC Conformance Claim

49 This protection profile claims conformance to

- Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model; Version 3.1, Revision 4, September 2012, CCMB-2012-09-001 [1]
- Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Components, Version 3.1, Revision 4, September 2012, CCMB-2012-09-002 [2]
- Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Requirements; Version 3.1, Revision 4, September 2012, CCMB-2012-09-003 [3]

as follows

- Part 2 conformant,
- Part 3 conformant.

The

- Common Methodology for Information Technology Security Evaluation, Evaluation Methodology; Version 3.1, Revision 4, September 2012, CCMB-2012-09-004, [4]

has to be taken into account.

## 2.2 Protection Profile Claim

50 This PP does not claim conformance to any protection profile.

## 2.3 Package Claim

51 The current PP is conformant to the following security requirements package:
Assurance package EAL5 augmented with AVA_VAN.5 as defined in the CC, part 3 [3].

## 2.4 Conformance Rationale

52 Since this PP does not claim conformance to any protection profile, this section is not applicable.

## 2.5 Conformance statement

53 This PP requires *strict* conformance of any ST or PP claiming conformance to this PP.

# Chapter 3 Security Problem Definition

## 3.1 Introduction

54 **Explanatory Note 9:** Some of the entities listed below, depending on context, can act both as an object to be protected (Section 3.1.1) as well as a subject (Section 3.1.2). Example: The SSP specifies that a user application may, for example, query itself. Thus, in FDP_ACC.2/AS.USER_PART_CONT (Section 106) the SSP is applied on the user application acting as object number 1 in Table 1 (Section 3.1.1.1) and on the (same) user application acting as subject number 1 in Table 3 (Section 3.1.2).

### *3.1.1 Assets and Objects*

55 Each partition, each communication object, and each system component consists of a triple: content, resources used by the content, and a shape, which contains a set of security attributes according to the SSP assigned to an entity linking content and resources (see Glossary for more details).

#### 3.1.1.1 Primary Assets

56 Primary assets represent user data.

| Object Number | Asset Name | Description, Operations | Generic Security Properties to be Maintained by the TOE, as long as the TOE is operational |
|---|---|---|---|
| 1 | User partition content (AS.USER_PART_CONT) | *User partition content* is user applications and/or data being executed and/or stored in a user partition.<br>This asset can be *executed* (user applications), and *stored* (user applications and data) by the TOE.<br>This asset can be *treated* (see Glossary) by subjects. | confidentiality, integrity |
| 2 | Communication object content (AS.COMMUN_OBJ_CONT) | *Communication object content* is the content of a communication object and exchanged (received and sent) between partitions.<br>This asset can be *exchanged* between partitions by the TOE.<br>This asset can be *treated* by subjects. | confidentiality, integrity |
| 3 | System component content (AS.SYS_COMP_CONT) | *System component content* is system applications and/or data being executed | confidentiality, integrity |

| Object Number | Asset Name | Description, Operations | Generic Security Properties to be Maintained by the TOE, as long as the TOE is operational |
|---|---|---|---|
| | | and/or stored in a system component (a system partition, a system extension or the on-board device support package).<br>This asset can be *executed* (system applications), and *stored* (system applications and data) by the TOE.<br>This asset can be *treated* by system applications. | |
| 4 | Audit data (AS.AUD) | *Audit data* – audit data is electronic records reflecting events to be audited.<br>This asset is *generated* by the TOE.<br>This asset can be *queried* by user applications and *treated* by system applications. | confidentiality, integrity |

Table 1: Primary Assets Representing User Data

#### 3.1.1.2 Secondary Assets

57 Secondary assets represent the TSF and TSF data.

| Object Number | Asset Name | Description, Operations | Generic Security Properties to be Maintained by the TOE, as long as the TOE is operational |
|---|---|---|---|
| 5 | User partition resources (AS.USER_PART_RES) | *User partition resources* comprise physical memory space, I/O memory space, a description of the set of CPUs the partition's applications can run on, allocated CPU time for each CPU, and interrupts. Resources are assigned according to the SSP.<br>This asset is *made available* or *made* | availability, confidentiality, integrity |

| Object Number | Asset Name | Description, Operations | Generic Security Properties to be Maintained by the TOE, as long as the TOE is operational |
|---|---|---|---|
| | | *unavailable* to user partitions by the TSF as configured according to the SSP. This asset is *used* by a user application being executed as a content of the related partition on its request.<br>This asset can be *treated* by system applications. | |
| 6 | User partition shape (AS.USER_PART_SHAPE) | A *user partition shape* contains a set of security attributes according to the SSP assigned to a user partition that links its *user partition resources* and its *user partition content*. A user partition shape contains, amongst other, an unambiguous partition identity, a flag indicating that the partition is a user partition, and the *resource usage data* (i.e. here partition resource usage data).<br>This asset is *used* by the TSF.<br>This asset can be *treated* by system applications.<br>For each instantiation of this object the TSF assigns a unique object identity (partition identity). | confidentiality, integrity |
| 7 | Communication object resources (AS.COMMUN_OBJ_RES) | *Communication object resources* are memory space. Resources are assigned according to the SSP.<br>This asset is *made available* or *made unavailable* to partitions by the TSF as configured according to the SSP. This asset is *used* by an application on its request.<br>This asset can be *treated* by system applications. | availability, confidentiality, integrity |
| 8 | Communication object shape (AS.COMMUN_OBJ_SHAPE) | A *communication object shape* contains a set of security attributes according to the SSP assigned to a communication object, which links its *communication* | confidentiality, integrity |

| Object Number | Asset Name | Description, Operations | Generic Security Properties to be Maintained by the TOE, as long as the TOE is operational |
|---|---|---|---|
| | | *object resources* and its *communication object content*. A communication object shape contains, amongst other, an unambiguous communication object identity and the *resource usage data* (i.e. here communication object resource usage data).<br>This asset is *used* by the TSF.<br>This asset can be *treated* by system applications.<br>For each instantiation of this object the TSF assigns a unique object identity (communication object identity). | |
| 9 | System component resources (AS.SYS_COMP_RES) | *Resources of a system component* comprise physical memory space, I/O memory space, a description of the set of CPUs the system component's applications can run on, for system partitions, allocated CPU time for each CPU, and interrupts.<br>Resources are assigned according to the SSP.<br>This asset is *made available* or *made unavailable* to system components by the TSF as configured according to the SSP. This asset is *used* by a system application being executed as a content of the related system partition, a system extension or a ODSP on its request.<br>This asset is *used* by a system component on its request. This asset can be *treated* by system applications. | availability, confidentiality, integrity |
| 10 | System component shape (AS.SYS_COMP_SHAPE) | A *system component shape* contains a set of security attributes according to the SSP assigned to a system component that links its *system component resources* and its *system component* | confidentiality, integrity |

| Object Number | Asset Name | Description, Operations | Generic Security Properties to be Maintained by the TOE, as long as the TOE is operational |
|---|---|---|---|
| | | *content*.<br>A system component shape of a system partition also contains, amongst other a flag indicating that the partition is a system partition, and the *resource usage data* (i.e. here partition resource usage data).<br>This asset is *used* by the TSF.<br>This asset can be *treated* by system applications.<br>For each instantiation of this object the TSF assigns a unique object identity (system component identity). | |
| 11 | Configuration data (AS.CONF_DATA) | *Configuration data* are data used by the TOE to enforce the SSP.<br>This asset is *stored* and *used* by the TSF.<br>This asset can be *treated* by system applications. | confidentiality, integrity |
| 12 | System application API (AS.SYS_APP_API) | The *system application API* is an interface to functions of the TSF available for system applications.<br>This asset is *made available* only to system applications and *made unavailable* to any user applications by the TSF according to the SSP.<br>This asset can be *invoked* by system applications. | availability (in the sense of ‘invokability’) only for system applications |

Table 2: Secondary Assets Representing the TSF and TSF Data

58 **Explanatory Note 10**: If a concrete TOE implementation cannot principally use any communication objects, the author of the related ST/PP shall clearly state it. In such a case the assets AS.COMMUN_OBJ_CONT, AS.COMMUN_OBJ_RES and AS.COMMUN_OBJ_SHAPE do not exist any more and, hence, should be ommited in all the related items like security objectives and security requirements.

The ability of the TOE to support system components and the generation of audit data is optional. If the TOE does not support system components or generate audit data, the assets AS.SYS_COMP_CONT, AS.COMMUN_OBJ_RES and AS.COMMUN_OBJ_SHAPE resp. AS.AUD do not exist any more and, hence, should be ommited in all the related items like security objectives and security requirements.

### 3.1.2 Subjects, Roles, and External Entities

| External Entity Number | Subject Number | Role | Definition |
|---|---|---|---|
| 1 | 1 | User application | A *user application* is any application within a user partition. A user application is allowed to use only the TOE user partition API.<br>For each instantiation of this subject the TOE assigns a unique subject identity. |
| 2 | 2 | System application | A *system application* is any application within a system partition, a system extension, or the on-board device support package (ODSP). Only a system application in a system partition is allowed to use the TOE system partition API. Only a system application in a system extension is allowed to use the TOE system extension API. Only a system application in the ODSP is allowed to use the TOE ODSP API.<br>For each instantiation of this subject the TOE assigns a unique subject identity. |
| 3 | - | System integrator | A *system integrator* is a person trusted to (re-)configure and integrate the TOE. This includes identifying system partitions and user partitions and assigning applications into partitions. |
| 4 | - | System operator | A *system operator* is a person trusted to (re-)install, stop, start, restart, or access (also physically) the TOE in the field. |
| 5 | - | Attacker | An attacker is a threat agent (a person or a process acting on his/her behalf) trying to undermine the TOE security policy defined by the current PP and, hence, the SSP. The attacker especially tries to change properties of the assets having to be maintained according to the TOE security policy defined by the current PP (see Table 1 and Table 2 in Section 3.1.1). The attacker is assumed to possess an at most high attack potential.<br>Note that the TOE security policy defined by the current PP only addresses attacks carried out by user applications and does not address any physical attacks, see P.SYSTEM_INTEGRATOR and P.SYSTEM_OPERATOR. All attacks from other sources than user applications shall be averted by the TOE operational environment. |

Table 3: Subjects, Roles and External Entities

59 In Table 3, if there is a number in the “subject” column, it means that, during operational use, the TSF recognizes the external entity as subject, and assigns a role to it. If there is no such number (“-”), then, during operational use, the TSF does not recognize that external entity as subject.

60 **Explanatory Note 11**: The ability of the TOE to support system components is optional. If the TOE does not support system components, the the role “System application” does not exist any more and, hence, should be ommited in all the related items like security objectives and security requirements.

## 3.2 Threats

61 Assets are defined in Table 1 in Section 3.1.1.1 (user data) and Table 2 in Section 3.1.1.2 (TSF data). An attacker is an external entity defined in Table 3 in Section 3.1.2.

### T.DISCLOSURE

62 An attacker discloses user data and/or TSF data of which the confidentiality shall be maintained according to Table 1 in Section 3.1.1.1 (user data) and Table 2 in Section 3.1.1.2 (TSF data).

### T.MODIFICATION

63 An attacker modifies user data and/or TSF data of which the integrity shall be maintained according to Table 1 in Section 3.1.1.1 (user data) and Table 2 in Section 3.1.1.2 (TSF data).

### T.DEPLETION

64 By requesting resources for communication objects and/or partitions and/or system extensions and/or ODSP, an attacker makes these resources unavailable to the TOE itself and/or to user applications and/or to system applications.

### T.EXECUTION

65 An attacker invokes a system application API without being authorized to do so.

66 **Explanatory Note 12**: For example, attacks can be initiated in the following ways:

- An arbitrary user application developer who, e.g. by subcontracting, is authorized to develop a user application for the TOE, tries to attack the TOE, e.g. to implant malicious code in the user application.
- An arbitrary external human entity or IT entity that has authorized access to a user application, e.g. from the Internet, compromises this user application to attack the TOE.

## 3.3 Organizational Security Policies

67 The TOE and/or its environment shall comply with the following organizational security policies (OSP) as security rules, procedures, practices or guidelines imposed by an organization upon its operation.

### P.AUDIT

68 The TOE shall be able to record all events to be audited as defined by the SSP.

69 **Explanatory Note 13**: The TOE enforces each possible SSP, i.e. a set of SSPs, concrete configuration parameters with their allowed values shall be exactly described in the TOE User Manuals.

### P.SAFE_SECURE_STATE

70 The TOE shall reach and keep a safe and secure state in which the TOE enforces the SSP.

### P.SYSTEM_INTEGRATOR

71 Obligations for a system integrator comprise, as follows:

(1) The system integrator shall select hardware such that:

(1.1) The hardware shall have CPU(s) with at least two privilege modes ("user" and "supervisor" mode).

**Explanatory Note 14:** Only the TOE separation kernel itself and system components may run in the "supervisor" mode. User applications always run in "user mode". In "user mode" only a limited set of instructions is available, in the "supervisor mode" all instructions are available.

(1.2) The hardware shall have memory management, which restricts accesses of user applications to memory regions according to the SSP.

**Explanatory Note 15**: Memory management can, for example, be provided by an MMU or a MPU.

(1.3) The hardware (CPU or CPUs) shall provide instructions to switch between privilege modes and to use the mamory management to set up different segments of memory.

(1.4) The hardware (CPU or CPUs) shall allow the TOE to reuse CPU(s) for different user applications, in a way that there is no residual information flow through CPU registers.

(1.5) The hardware shall provide default values for security-relevant settings at power-on (e.g. program counter, a full list shall be included in the TOE User Manuals).

**Explanatory Note 16**: This supports the TOE reaching the initial safe and secure state.

(1.6) If the hardware possesses any other active components beside CPUs, then either the hardware shall provide support to either turn these components completely off or the TOE separation kernel and/or system components control them as described in TOE User Manuals.

**Explanatory Note 17**: For example, if devices can execute DMA, then all DMA shall be switched off or, in order to control DMA, the hardware shall provide an I/O MMU, with the I/O MMU controlled by the TOE separation kernel and/or system components.

**Application Note 1**: The writer of a ST shall state all the CPU architectures which should be subject of consideration during the security evaluation. These architectures shall fulfill requirements (1.1) to (1.3). Depending on the system integrator's requirements for residual information flow on the hardware, special attention may have to be paid to (1.4) to (1.6).

(2) The system integrator shall ensure that the TOE separation kernel gets exclusively invoked, so that the TSF starts operating exclusively controlling the CPU(s) and other hardware resources it has to control. For this reason, the system integrator shall ensure an appropriate implementation and configuration firmware and bootloader and ODSP.

(3) The system integrator shall select timer facilities according to the SIP.

(4) The system integrator shall ensure that any system component content has been developed following the guidance in the TOE User Manuals and enables enforcing the SSP during operational use. The system integrator shall approve the system component content for integration.

(5) The system integrator shall correctly perform the integration process according to the guidance in the TOE User Manuals.
The system integrator is fully responsible for the definition of an appropriate – for the purpose of the system integrator – System Security Policy (SSP). The TSF will enforce any SSP as defined by the system integrator.

(6) The system integrator shall define an operational policy for the product in the field which, amongst other, enables enforcing the SSP during operational use. The system integrator shall oblige the system operator to follow this policy. The operational policy shall at least require that:

(6.1) The system operator shall ensure that the operational environment provides the TOE with appropriate physical security measures commensurate with the value and properties of the assets protected by the TOE.

(6.2) The system operator shall ensure that the hardware selected for the TOE operates correctly according to the operational policy (and, if necessary, according to the hardware manuals).

### P.SYSTEM_OPERATOR

72 The system operator shall follow the operational policy for the product in the field defined by the system integrator..

## 3.4 Assumptions

73 This section describes the assumptions about the operational environment of the TOE.

74 **A.TRUSTWORTHY_PERSONNEL**

The personnel configuring and integrating the TOE (system integrator) are trustworthy, act according to Section 3.3, organizational security policy P.SYSTEM_INTEGRATOR and are sufficiently qualified for this task.

The personnel installing and operating the TOE (system operator) are trustworthy, act according to Section 3.3, organizational security policy P.SYSTEM_OPERATOR and are sufficiently qualified for this task.

# Chapter 4 Security Objectives

## 4.1 Security Objectives for the TOE

75 **OT.AUDIT**

The TOE shall be able to record all events to be audited as defined by the SSP.

76 **OT.CONFIDENTIALITY**

For each asset, the TOE shall preserve its confidentiality according to Table 1 in Section 3.1.1.1 (user data) and Table 2 in Section 3.1.1.2 (TSF data).

77 **OT.INTEGRITY**

For each asset, the TOE shall preserve its integrity according to Table 1 in Section 3.1.1.1 (user data) and Table 2 in Section 3.1.1.2 (TSF data).

78 **OT.RESOURCE_AVAILABILITY**

The TOE shall ensure the availability of partition resources, communication object resources and system component resources at their request.

79 **OT.SAFE_SECURE_STATE**

The TOE shall reach and keep a safe and secure state. A safe and secure state is a TOE state in which the TOE enforces the SSP.

80 **OT.SYSTEM_APPLICATION_API_PROTECTION**

The TOE shall prevent any invocation of the system application API by a user application.

## 4.2 Security Objectives for the Operational Environment

81 **OE.SYSTEM_INTEGRATOR**

Obligations for a system integrator comprise, as follows:

(1) The system integrator shall select hardware such that:

(1.1) The hardware shall have CPU(s) with at least two privilege modes (“user” and “supervisor” mode).

(1.2) The hardware shall have memory management, which restricts accesses of user applications to memory regions according to the SSP.

Explanatory Note 15: Memory management can, for example, be provided by an MMU or a MPU.

(1.3) The hardware (CPU or CPUs) shall provide instructions to switch between privilege modes and to use the mamory management to set up different segments of memory.

(1.4) The hardware (CPU or CPUs) shall allow the TOE to reuse CPU(s) for different user applications, in a way that there is no residual information flow through CPU registers. (1.5) The hardware shall provide default values for security-relevant settings at power-on (e.g. program counter, a full list shall be included in the TOE User Manuals).

(1.6) If the hardware possesses any other active components beside CPUs, then either the hardware shall provide support to either turn these components completely off or the TOE separation kernel and/or system components control them as described in TOE User Manuals.

(2) The system integrator shall ensure that the TOE separation kernel gets exclusively invoked, so that the TSF starts operating exclusively controlling the CPU(s) and other hardware resources it has to control. For this reason, the system integrator shall ensure an appropriate implementation and configuration firmware and bootloader and ODSP.

(3) The system integrator shall select timer facilities according to the SIP.

(4) The system integrator shall ensure that any system component content has been developed following the guidance in the TOE User Manuals and enables enforcing the SSP during operational use. The system integrator shall approve the system component content for integration.

(5) The system integrator shall correctly perform the integration process according to the guidance in the TOE User Manuals.
The system integrator is fully responsible for the definition of an appropriate – for the purpose of the system integrator – System Security Policy (SSP). The TSF will enforce any SSP as defined by the system integrator.

(6) The system integrator shall define an operational policy for the product in the field which, amongst other, enables enforcing the SSP during operational use. The system integrator shall oblige the system operator to follow this policy. The operational policy shall at least require that:

(6.1) The system operator shall ensure that the operational environment provides the TOE with appropriate physical security measures commensurate with the value and properties of the assets protected by the TOE.

(6.2) The system operator shall ensure that the hardware selected for the TOE operates correctly according to the operational policy (and, if necessary, according to the hardware manuals).

82 **OE.SYSTEM_OPERATOR**

The system operator shall follow the operational policy for the product in the field defined by the system integrator.

83 **OE.TRUSTWORTHY_PERSONNEL**

The personnel configuring and integrating the TOE (system integrator) are trustworthy, act according to Section 3.3, organizational security policy P.SYSTEM_INTEGRATOR and are sufficiently qualified for this task.
The personnel installing and operating the TOE (system operator) are trustworthy, act according to Section 3.3, organizational security policy P.SYSTEM_OPERATOR and are sufficiently qualified for this task.

## 4.3 Security Objectives Rationales

84 The following table provides an overview for security objectives coverage (TOE and its environment) and also gives an evidence for sufficiency and necessity of the defined objectives. It shows that all threats and OSPs are addressed by the security objectives and it also shows that all assumptions are addressed by the security objectives for the TOE operational environment.

| | OT.CONFIDENTIALITY | OT.INTEGRITY | OT.RESOURCE_AVAILABILITY | OT.SYSTEM_APPLICATION_API_PROTECTION | OT.AUDIT | OT.SAFE_SECURE_STATE | OE.SYSTEM_INTEGRATOR | OE.SYSTEM_OPERATOR | OE.TRUSTWORTHY_PERSONNEL |
|---|---|---|---|---|---|---|---|---|---|
| T.DISLOSURE | X | | | | | | | | |
| T.MODIFICATION | | X | | | | | | | |
| T.DEPLETION | | | X | | | | | | |
| T.EXECUTION | | | | X | | | | | |
| P.AUDIT | | | | | X | | | | |
| P.SAFE_SECURE_STATE | | | | | | X | | | |
| P.SYSTEM_INTEGRATOR | | | | | | | X | | |
| P.SYSTEM_OPERATOR | | | | | | | | X | |
| A.TRUSTWORTHY_PERSONNEL | | | | | | | | | X |

Table 4: Security Objectives Rationale

85 A justification required for *suitability* of the security objectives to cope with the security problem definition is given below:

### *4.3.1 Security Objective Rationales: Threats*

#### 4.3.1.1 Threat: T.DISCLOSURE

86 If the security objective OT.CONFIDENTIALITY has been reached, the threat T.DISCLOSURE is completely eliminated.

#### 4.3.1.2 Threat: T.MODIFICATION

87 If the security objective OT.INTEGRITY has been reached, the threat T.MODIFICATION is completely eliminated.

#### 4.3.1.3 Threat: T.DEPLETION

88 If the security objective OT.RESOURCE_AVAILABILITY has been reached, the threat T.DEPLETION is completely eliminated.

#### 4.3.1.4 Threat: T.EXECUTION

89 If the security objective OT.SYSTEM_APPLICATION_API_PROTECTION has been reached, the threat T.EXECUTION is completely eliminated.

### *4.3.2 Security Objective Rationales: Security Policies*

90 Each identified security policy in this Protection Profile is addressed by at least one security objective for the TOE or security objective for the operational environment. This section provides a mapping from each security policy to the security objectives and provides a rationale how the security policy is fulfilled.

#### 4.3.2.1 Policy: P.AUDIT

91 OT.AUDIT directly enforces P.AUDIT.

#### 4.3.2.2 Policy: P.SAFE_SECURE_STATE

92 OT.SAFE_SECURE_STATE directly enforces P.SAFE_SECURE_STATE.

#### 4.3.2.3 Policy P.SYSTEM_INTEGRATOR

93 OE.SYSTEM_INTEGRATOR directly enforces P.SYSTEM_INTEGRATOR.

#### 4.3.2.4 Policy: P.SYSTEM_OPERATOR

94 OE.SYSTEM_OPERATOR directly enforces P.SYSTEM_OPERATOR.

### *4.3.3 Security Objective Rationales: Assumptions*

95 Each security assumption in this Protection Profile is addressed by at least one security objective for the operational environment. This section maps assumptions to environmental security objectives and provides a rationale how the assumption is fulfilled.

#### 4.3.3.1 Assumption: A.TRUSTWORTHY_PERSONNEL

96 OE.TRUSTWORTHY_PERSONNEL directly upholds A.TRUSTWORTHY_PERSONNEL.

# Chapter 5 Extended Components Definition

97 This PP does not include any extended components.

# Chapter 6 Security Requirements

98 This part of the PP defines the detailed security requirements that shall be satisfied by the TOE. The statement of TOE security requirements shall define the *functional* and *assurance* security requirements that the TOE needs to satisfy in order to meet the security objectives for the TOE.

99 The CC allows several operations to be performed on security requirements (on the component level); *refinement*, *selection*, *assignment* and *iteration* are defined in sec. 8.1 of Part 1 [1] of the CC. Each of these operations is used in this PP.

100 The **refinement** operation is used to add detail to a requirement, and, thus, further restricts a requirement. Refinements of security requirements are denoted in such a way that added words are in underlined and removed words are ~~crossed out~~.

101 The **selection** operation is used to select one or more options provided by the CC in stating a requirement. Selections having been made by the PP author are *italicised*. Selections to be filled in by the ST author appear in square brackets with an indication that a selection has to be made, [selection:], and are *italicised*.

102 The **assignment** operation is used to assign a specific value to an unspecified parameter, such as the length of a password. Assignments having been made by the PP author are denoted by showing as **bold** text. Assignments to be filled in by the ST author appear in square brackets with an indication that an assignment has to be made [assignment:], and are *italicised*. In some cases the assignment made by the PP authors defines a selection to be performed by the ST author. Thus this text is underlined and italicised like ***this***.

103 The **iteration** operation is used when a component is repeated with varying operations. Iteration is denoted by showing a slash "/", and the iteration indicator after the component identifier. For example, FDP_ACF.1/AS.USER_PART_CONT indicates an iteration of FDP_ACF.1 on the asset 'user partition content'. Iterations applied to assets follow the order of Table 1 in Section 3.1.1.1 (primary assets) and Table 2 in Section 3.1.1.2 (secondary assets). For the sake of a better readability, the iteration operation may also be applied to some single components (being not repeated) in order to indicate their relation to other SFRs with the same iteration indicator. In such a case, the iteration operation is applied to only one single component.

## 6.1 Security Functional Requirements for the TOE

### *6.1.1 Overview*

104 In order to give an overview of the SFRs in the context of the security services offered by the TOE, in the following table the authors of this PP defined security functional groups and allocated the functional requirements described in the following sections to them.

| Security Functional Group | Security Functional Requirements (SFRs)<br>(SFRs always used together are grouped by "{}") |
|---|---|
| SFG_SSA: Separation in space of applications hosted in different partitions from each other and from the TOE operating system | {FDP_ACC.2/AS.USER_PART_CONT, FDP_ACF.1/AS.USER_PART_CONT}, {FDP_ACC.2/AS.SYS_COMP_CONT, FDP_ACF.1/AS.SYS_COMP_CONT}, {FDP_IFC.2, FDP_IFF.1}, FDP_IFF.5, FRU_RSA.2/AS.USER_PART_RES, FRU_RSA.2/AS.SYS_COMP_RES<br>Supported by:<br>The entire class FMT (except for FMT_MTD.1/AS.COMMUN_OBJ_RES, FMT_MTD.1/AS.COMMUN_OBJ_SHAPE), the entire class FPT |
| SFG_STA: Separation in time of applications hosted in different partitions from each other and from the TOE operating system | {FDP_IFC.2, FDP_IFF.1}, FDP_IFF.5, FDP_RIP.2, FRU_PRS.1, FRU_RSA.2/AS.USER_PART_RES, FRU_RSA.2/AS.SYS_COMP_RES<br>Supported by:<br>The entire class FMT (except for FMT_MTD.1/AS.COMMUN_OBJ_RES, FMT_MTD.1/AS.COMMUN_OBJ_SHAPE), the entire class FPT |
| SFG_COM: Provision and management of communication objects | {FDP_ACC.2/AS.COMMUN_OBJ_CONT, FDP_ACF.1/AS.COMMUN_OBJ_CONT}, {FDP_IFC.2, FDP_IFF.1}, FDP_IFF.5, FRU_RSA.2/AS.COMMUN_OBJ_RES<br>Supported by:<br>FMT_MTD.1/AS.COMMUN_OBJ_RES, FMT_MTD.1/AS.COMMUN_OBJ_SHAPE |
| SFG_MAN: Management of and access to the TSF and TSF data | FIA_UID.2, all selected components of the class FMT |
| SFG_SPT: TSF self-protection and accuracy of security functionality | FPT_FLS.1, FPT_RCV.2<br>Supported by:<br>FIA_UID.2, the entire class FMT |
| SFG_AUD: Generation and treatment of audit data according to the SSP | FAU_GEN.1, {FDP_ACC.2/AS.AUD, FDP_ACF.1/AS.AUD}<br>Supported by:<br>FIA_UID.2, FMT_MOF.1, FMT_SMF.1, FMT_MSA.1, FMT_MSA.3, FMT_SMR.1, the entire class FPT |

Table 5: Security Functional Groups and their SFRs

### 6.1.2 Class FAU Security Audit

#### 6.1.2.1 FAU_GEN.1 Audit Data Generation

Hierarchical to: No other components.

Dependencies: FPT_STM.1: not fulfilled, but justified: reliable timestamps shall be provided to the TOE by the hardware platform as enforced by **P.SYSTEM_INTEGRATOR**.

**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) All auditable events for the [selection: *minimum, basic, detailed, not specified*] level of audit; and

c) **All events to be audited as defined by the SSP**[1].

**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, subject identity (if applicable), and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [assignment: *other audit relevant infomation*].

Explanatory Note 18: A conformant ST/PP can specify to have no audit functionality at all by selecting the following formulation for FAU_GEN.1.1: “The TSF does **not** provide functionality to generate an audit record.”

### 6.1.3 Class FDP User Data Protection

105 Objects (user data assets) are defined in Table 1 in Section 3.1.1.1. Subjects are defined in Table 3 in Section 3.1.2. For the security attributes “asset” see column “Asset Name” in Table 1, for “object identity” see Table 2, for “role” and “subject identity” see Table 3. The set of all operations among subjects and objects is defined in Table 1 in Section 3.1.1.1, column “Description, Operations”.

#### 6.1.3.1 FDP_ACC.2 Complete Access Control

106 **FDP_ACC.2/AS.USER_PART_CONT for Asset: ‘User Partition Content’ as Object**

Hierarchical to: FDP_ACC.1

[1] [assignment: *other specifically defined auditable events*]

Dependencies: FDP_ACF.1: fulfilled by FDP_ACF.1/AS.USER_PART_CONT

**FDP_ACC.2.1** The TSF shall enforce **the System Security Policy (SSP)**[2] on **all subjects and 'user partition content' as object**[3] and all operations among subjects and objects covered by the SFP.

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

### 107 FDP_ACC.2/AS.COMMUN_OBJ_CONT for Asset: 'Communication Object Content' as Object

Hierarchical to: FDP_ACC.1

Dependencies: FDP_ACF.1: fulfilled by FDP_ACF.1/AS.COMMUN_OBJ_CONT.

**FDP_ACC.2.1** The TSF shall enforce **the System Security Policy (SSP)**[4] on **all subjects and 'communication object content' as object**[5] and all operations among subjects and objects.

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

### 108 FDP_ACC.2/AS.SYS_COMP_CONT for Asset: 'System Component Content' as Object

Hierarchical to: FDP_ACC.1

Dependencies: FDP_ACF.1: fulfilled by FDP_ACF.1/AS.SYS_COMP_CONT

**FDP_ACC.2.1** The TSF shall enforce **the System Security Policy (SSP)**[6] on **all subjects and 'system component content' as object**[7] and all operations among subjects and objects.

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

[2] [assignment: *access control SFP*]
[3] [assignment*: list of subjects and object*]
[4] [assignment: *access control SFP*]
[5] [assignment: *list of subjects and object*]
[6] [assignment: *access control SFP*]
[7] [assignment: *list of subjects and object*]

109 **FDP_ACC.2/AS.AUD for Asset: 'Audit Data' as Object**

| | |
|---|---|
| Hierarchical to: | FDP_ACC.1 |
| Dependencies: | FDP_ACF.1: fulfilled by FDP_ACF.1/AS.AUD |
| **FDP_ACC.2.1** | The TSF shall enforce **the System Security Policy (SSP)**[8] on **all subjects and 'audit data' as object**[9] and all operations among subjects and objects. |
| **FDP_ACC.2.2** | The TSF shall ensure that all operations between any subject controlled by the TSF and any object controlled by the TSF are covered by an access control SFP. |

### 6.1.3.2 FDP_ACF.1 Access Control Functions

110 **FDP_ACF.1/AS.USER_PART_CONT for Asset: 'User Partition Content' as Object**

| | |
|---|---|
| Hierarchical to: | No other components. |
| Dependencies: | FDP_ACC.1: hierarchically fulfilled by FDP_ACC.2/AS.USER_PART_CONT; FMT_MSA.3: fulfilled by FMT_MSA.3. |
| **FDP_ACF.1.1** | The TSF shall enforce the **SSP**[10] to objects based on the following: **the subjects and objects defined in Section 3.1 and the respective security subject attributes "role", "subject identity" and object security attributes "asset", "object identity"**[11]. |
| **FDP_ACF.1.2** | The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: **a subject with the attribute "role" set to "user application" is allowed to treat the object with attribute "asset" set to "user partition content", if and only if the "subject identity" is in the "user partition shape" linked to the "user partition content"**[12]. |

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

[9] [assignment: *list of subjects and object*]

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

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

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

**FDP_ACF.1.3** The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: **the subject with the attribute "role" set to "system application" is always allowed to treat the object with attribute "asset" set to "user partition content"**[13].

**FDP_ACF.1.4** The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [assignment: *rules, based on security attributes, that explicitly deny access of subjects to objects*].

111 **FDP_ACF.1/AS.COMMUN_OBJ_CONT for Asset: 'Communication Object Content' as Object**

Hierarchical to: No other components.

Dependencies: FDP_ACC.1: hierarchically fulfilled by FDP_ACC.2/AS.COMMUN_OBJ_CONT; FMT_MSA.3: fulfilled by FMT_MSA.3.

**FDP_ACF.1.1** The TSF shall enforce the **SSP**[14] to objects based on the following: **the subjects and objects defined in Section 3.1 and the respective security subject attributes "role", "subject identity" and object security attributes "asset", "object identity"**[15].

**FDP_ACF.1.2** The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: **A subject with the attribute "role" set to "user application" is allowed to treat the object with attribute "asset" set to "communication object content", if and only if the attribute "subject identity" and "object identity" have values for which the SSP allows treating this object by this subject**[16].

**FDP_ACF.1.3** The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: **the subject with the attribute "role" set to "system application" is always allowed to treat the object with attribute "asset" set to "communication object content"**[17].

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

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

[15] [assignment: *list of subjects and object*]

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

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

**FDP_ACF.1.4** The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [assignment: *rules, based on security attributes, that explicitly deny access of subjects to objects*]**.**

### 112 FDP_ACF.1/AS.SYS_COMP_CONT for Asset: ‘System Component Content’ as Object

Hierarchical to: No other components.

Dependencies: Dependencies: FDP_ACC.1: hierarchically fulfilled by FDP_ACC.2/AS.SYS_COMP_CONT; FMT_MSA.3: fulfilled by FMT_MSA.3.

**FDP_ACF.1.1** The TSF shall enforce the **SSP**[18] to objects based on the following: **the subjects and objects defined in Section 3.1 and the respective security subject attribute “role” and object security attribute “asset”**[19].

**FDP_ACF.1.2** The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: **a subject with the attribute “role” set to “user application” is not allowed to treat the object with attribute “asset” set to “system component content”**[20].

**FDP_ACF.1.3** The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: **the subject with the attribute “role” set to “system application” is always allowed to treat the object with attribute “asset” set to “system component content”**[21].

**FDP_ACF.1.4** The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [assignment: *rules, based on security attributes, that explicitly deny access of subjects to objects*].

### 113 FDP_ACF.1/AS.AUD for Asset: ‘Audit Data’ as Object

Hierarchical to: No other components

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

[19] [assignment: *list of subjects and object*]

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

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

Dependencies: FDP_ACC.1: hierarchically fulfilled by FDP_ACC.2/AS.AUD; FMT_MSA.3: fulfilled by FMT_MSA.3.

**FDP_ACF.1.1** The TSF shall enforce the **SSP**[22] to objects based on the following: **the subjects and objects defined in Section 3.1 and the respective security subject attributes “role”, “subject identity” and object security attribute “asset”**[23].

**FDP_ACF.1.2** The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: **A subject with the attribute “role” set to “user application” is allowed to query the object with attribute “asset” set to “audit data”, if and only if the attribute “subject identity” has a value for which the SSP allows reading audit data**[24].

**FDP_ACF.1.3** The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: **the subject with the attribute “role” set to “system application” is always allowed to query the object with attribute “asset” set to “audit data”**[25].

**FDP_ACF.1.4** The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [assignment: *rules, based on security attributes, that explicitly deny access of subjects to objects*].

### 6.1.3.3 FDP_IFC.2 Complete Information Flow Control

Hierarchical to: FDP_IFC.1

Dependencies: FDP_IFF.1: fulfilled by FDP_IFF.1.

**FDP_IFC.2.1** The TSF shall enforce the **Partitioned Information Flow Policy (PIFP)**[26] on

- **all subjects;**
- **all objects**[27]

and all operations that cause that information to flow to and from subjects covered by the SFP.

[22] [assignment: *access control SFP*]
[23] [assignment: *list of subjects and object*]
[24] [assignment: *rules governing access among controlled subjects and controlled objects using controlled operations on controlled objects*]
[25] [assignment: *rules, based on security attributes, that explicitly authorise access of subjects to objects*]
[26] [assignment: *information flow control SFP*]
[27] [assignment: *list of subjects and information*]

**FDP_IFC.2.2** The TSF shall ensure that all operations that cause any information in the TOE to flow to and from any subject in the TOE are covered by an information flow control SFP.

### 6.1.3.4 FDP_IFF.1 Simple Security Attributes

Hierarchical to: No other components.

Dependencies: FDP_IFC.1: hierarchically fulfilled by FDP_IFC.2; FMT_MSA.3: not iterated for PIFP by FMT_MSA.3, but justified: As PIFP is derived from SSP, FMT_MSA.3 for SSP implies static policy initialization and management of security attributes also for PIFP.

**FDP_IFF.1.1** The TSF shall enforce the **PIFP**[28] based on the following types of subject and information security attributes:

- **subject security attributes "role" and "subject identity";**
- **object security attribute "asset", "object identity";**
- **type of operation**

**as defined in Section 3.1**[29].

**FDP_IFF.1.2** The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold: **The operation is allowed by the SSP**[30].

**FDP_IFF.1.3** The TSF shall enforce the additional information flow rules: [assignment: *additional information flow control SFP rules*].

**FDP_IFF.1.4** The TSF shall explicitly authorize an information flow based on the following rules: [assignment: *rules, based on security attributes, that explicitly authorise information flows*].

**FDP_IFF.1.5** The TSF shall explicitly deny an information flow based on the following rules: [assignment: *rules, based on security attributes, that explicitly deny information flows*].

### 6.1.3.5 FDP_IFF.5 No Illicit Information Flows

Hierarchical to: FDP_IFF.4

[28] [assignment: *information flow control SFP*]

[29] [assignment: *list of subjects and information controlled under the indicated SFP, and for each, the security attributes*]

[30] [assignment: *for each operation, the security attribute-based relationship that must hold between subject and information security attributes]*

Dependencies: FDP_IFC.1, hierarchically fulfilled by FDP_IFC.2.

**FDP_IFF.5.1** The TSF shall ensure that no illicit information flows exist to circumvent **the PIFP**.[31]

### 6.1.3.6 FDP_RIP.2 Full Residual Information Protection

Hierarchical to: FDP_RIP.1

Dependencies: No dependencies.

**FDP_RIP.2.1** The TSF shall ensure that any previous information content of ~~a resource~~ all CPU registers and memory caches being relevant to a partition switch, [assignment: *list of other resources*] is made unavailable upon the [selection: *allocation of the resource to, deallocation of the resource from*] all objects.

114 Explanatory Note 19: Partition switches are defined by SSP as part of the scheduling scheme.

## *6.1.4 Class FIA Identification and Authentication*

### 6.1.4.1 FIA_UID.2 User Identification

Hierarchical to: FIA_UID.1

Dependencies: No dependencies.

**FIA_UID.2.1** The TSF shall require each ~~user~~ application to be successfully identified before allowing any other TSF-mediated actions on behalf of that ~~user~~ application.

115 Explanatory Note 20: A “user” of the TOE is a user application or a system application.

## *6.1.5 Class FMT Security Management*

### 6.1.5.1 FMT_MOF.1 Management of Security Functions Behavior

Hierarchical to: No other components.

Dependencies: FMT_SMF.1, fulfilled by FMT_SMF.1; FMT_SMR.1, fulfilled by FMT_SMR.1.

**FMT_MOF.1.1** The TSF shall restrict the ability to *invoke*[32] the functions **identified in FMT_SMF.1**[33] to **system applications**[34].

[31] [assignment: *name of information flow control SFP*].

[32] [selection: *determine the behaviour of, disable, enable, modify the behaviour of*] refinement

[33] [assignment: *list of functions*]

### 6.1.5.2 FMT_MSA.1 Management of Security Attributes

Hierarchical to: No other components.

Dependencies: [FDP_ACC.1, hierarchically fulfilled by FDP_ACC.2/AS.COMMUN_OBJ_CONT, FDP_ACC.2/AS.USER_PART_CONT, FDP_ACC.2/AS.SYS_COMP_CONT, and FDP_ACC.2/AS.AUD; and FDP_IFC.1: hierarchically fulfilled by FDP_IFC.2]; FMT_SMF.1: fulfilled by FMT_SMF.1; FMT_SMR.1: fulfilled by FMT_SMR.1.

**FMT_MSA.1.1** The TSF shall enforce the **SSP**[35] to restrict the ability to *modify*[36], [selection: *change_default, query, delete, [assignment: other operations]*] the security attributes **role, asset, subject identity, and object identity**[37] to **no one**[38].

### 6.1.5.3 FMT_MSA.3 Static Policy Attribute Initialization

Hierarchical to: No other components.

Dependencies: FMT_MSA.1: fulfilled by FMT_MSA.1, FMT_SMR.1: fulfilled by FMT_SMR.1.

**FMT_MSA.3.1** The TSF shall enforce the **SSP**[39] to provide [selection, choose one of: *restrictive, permissive, [assignment: other property]*] default values for security attributes that are used to enforce the SFP.

**FMT_MSA.3.2** The TSF shall allow the [assignment: *the authorised identified roles*] to specify alternative initial values to override the default values when an object or information is created.

116 Explanatory Note 21: Default and alternative initial values for security attributes used to enforce the SSP as well as the related authorised identified roles should be appropriate for this purpose.

### 6.1.5.4 FMT_MTD.1 Management of TSF Data

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

[35] [assignment: *access control SFP(s), information flow control SFP(s)*]

[36] [selection: *change_default, query, modify, delete, [assignment: other operations]*]

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

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

[39] [assignment: *access control SFP, information flow control SFP*]

### 117 FMT_MTD.1/AS.USER_PART_RES for Asset: ‘User Partition Resources’

Hierarchical to: No other components.

Dependencies: FMT_SMF.1: not fulfilled by FMT_SMF.1, but justified: a related management function is implemented not in the TSF itself, but can be implemented in a system application; FMT_SMR.1: fulfilled by FMT_SMR.1.

**FMT_MTD.1.1** The TSF shall restrict the ability to ***treat***[40] the **‘user partition resources’**[41] to **system applications**[42].

### 118 FMT_MTD.1/AS.USER_PART_SHAPE for Asset: ‘User Partition Shape’

Hierarchical to: No other components.

Dependencies: FMT_SMF.1: not fulfilled by FMT_SMF.1, but justified: a related management function is implemented not in the TSF itself, but can be implemented in a system application; FMT_SMR.1: fulfilled by FMT_SMR.1.

**FMT_MTD.1.1** The TSF shall restrict the ability to ***treat***[43] the **‘user partition shapes**[44]**’** to **system applications**[45]**.**

### 119 FMT_MTD.1/AS.COMMUN_OBJ_RES for Asset: ‘Communication Object Resources’

Hierarchical to: No other components.

Dependencies: FMT_SMF.1: not fulfilled by FMT_SMF.1, but justified: a related management function is implemented not in the TSF itself, but can be implemented in a system application; FMT_SMR.1: fulfilled by FMT_SMR.1.

**FMT_MTD.1.1** The TSF shall restrict the ability to ***treat***[46] the **‘communication object resources’**[47] to **system applications**[48].

[40] [selection: *change_default, query, modify, delete, clear, [assignment: other operations]*]
[41] [assignment: *list of TSF data*]
[42] [assignment: *the authorised identified roles*]
[43] [selection: *change_default, query, modify, delete, clear, [assignment: other operations]*]
[44] [assignment: *list of TSF data*]
[45] [assignment: *the authorised identified roles*]
[46] [selection: *change_default, query, modify, delete, clear, [assignment: other operations]*]
[47] [assignment: *list of TSF data*]
[48] [assignment: *the authorised identified roles*]

### 120 FMT_MTD.1/AS.COMMUN_OBJ_SHAPE for Asset: ‘Communication Object Shape’

Hierarchical to: No other components.

Dependencies: FMT_SMF.1: not fulfilled by FMT_SMF.1, but justified: a related management function is implemented not in the TSF itself, but can be implemented in a system application; FMT_SMR.1: fulfilled by FMT_SMR.1.

**FMT_MTD.1.1** The TSF shall restrict the ability to ***treat***[49] the **‘communication object shapes’**[50] to **system applications**[51]**.**

### 121 FMT_MTD.1/AS.SYS_COMP_RES for Asset: ‘System Component Resource’

Hierarchical to: No other components.

Dependencies: FMT_SMF.1: not fulfilled by FMT_SMF.1, but justified: a related management function is implemented not in the TSF itself, but can be implemented in a system application; FMT_SMR.1: fulfilled by FMT_SMR.1.

**FMT_MTD.1.1** The TSF shall restrict the ability to ***treat***[52] the ‘**system component resources**’[53] to **system applications**[54].

### 122 FMT_MTD.1/AS.SYS_COMP_SHAPE for Asset: ‘System Component Shape’

Hierarchical to: No other components.

Dependencies: FMT_SMF.1: not fulfilled by FMT_SMF.1, but justified: a related management function is implemented not in the TSF itself, but can be implemented in a system application; FMT_SMR.1: fulfilled by FMT_SMR.1.

**FMT_MTD.1.1** The TSF shall restrict the ability to ***treat***[55] the ‘**system component shapes**’[56] to **system applications**[57].

[49] [selection: *change_default, query, modify, delete, clear, [assignment: other operations]*]
[50] [assignment: *list of TSF data*]
[51] [assignment: *the authorised identified roles*]
[52] [selection: *change_default, query, modify, delete, clear, [assignment: other operations]*]
[53] [assignment: *list of TSF data*]
[54] [assignment: *the authorised identified roles*]
[55] [selection: *change_default, query, modify, delete, clear, [assignment: other operations]*]
[56] [assignment: *list of TSF data*]
[57] [assignment: *the authorised identified roles*]

123 **FMT_MTD.1/AS.CONF_DATA for Asset: ‘Configuration Data’**

Hierarchical to: No other components.

Dependencies: FMT_SMF.1: not fulfilled by FMT_SMF.1, but justified: a related management function is implemented not in the TSF itself, but can be implemented in a system application; FMT_SMR.1: fulfilled by FMT_SMR.1.

**FMT_MTD.1.1** The TSF shall restrict the ability to ***treat***[58] the **‘configuration data’**[59] to **system applications**[60].

124 **FMT_MTD.1/AS.SYS_APP_API for Asset: ‘System Application API’**

Hierarchical to: No other components.

Dependencies: Dependencies: FMT_SMF.1: not fulfilled by FMT_SMF.1, but justified: a related management function is implemented not in the TSF itself, but can be implemented in a system application; FMT_SMR.1: fulfilled by FMT_SMR.1.

**FMT_MTD.1.1** The TSF shall restrict the ability to ***invoke***[61] the **‘System Application API’**[62] to **system applications**[63].

### 6.1.5.5 FMT_SMF.1 Specification of Management Functions

Hierarchical to: No other components.

Dependencies: No dependencies.

**FMT_SMF.1.1** The TSF shall be capable of performing the following management functions: [assignment: *list of management functions to be provided by the TSF*].

125 Explanatory Note 22: For example, en- / disabling the audit function.

[58] [selection: *change_default, query, modify, delete, clear, [assignment: other operations]*]
[59] [assignment: *list of TSF data*]
[60] [assignment: *the authorised identified roles*]
[61] [selection: *change_default, query, modify, delete, clear, [assignment: other operations]*]
[62] [assignment: *list of TSF data*]
[63] [assignment: *the authorised identified roles*]

### FMT_SMR.1 Security Roles

Hierarchical to: No other components.

Dependencies: FIA_UID.1, hierarchically fulfilled by FIA_UID.2.

**FMT_SMR.1.1** The TSF shall maintain the roles:

- **“system application” and**
- **“user application”**[64].
- [assignment: *list of further* *authorised identified roles compliant with* Table 3].

**FMT_SMR.1.2** The TSF shall be able to associate ~~users with roles~~ each application with a role.

## *6.1.6 Class FPT Protection of the TSF*

### FPT_FLS.1 Failure with Preservation of Secure State

Hierarchical to: No other components.

Dependencies: No dependencies.

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

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

126 Explanatory Note 23: An example for an instantiation of the list of types of failures may be “TOE initialization error”, “TOE run-time error”, “partition initialization error”, “partition run-time error”.

### FPT_RCV.2 Automated Recovery

Hierarchical to: FPT_RCV.1.

Dependencies: AGD_OPE.1: fulfilled by the assurance package chosen.

**FPT_RCV.2.1** When automated recovery from

- **TOE initialization**[65]
- [assignment: *list of further failures/service discontinuities*].

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

[65] [assignment: *list of failures/service discontinuities*]

is not possible, the TSF shall enter a halt state ~~a maintenance mode where the ability to return to a secure state is provided~~.

127 Explanatory Note 24: This element describes an early phase during initialization, where automated recovery as defined in FPT_RCV.2.2 is not yet possible, because the TSF's mechanism to handle errors is not yet active.

**FPT_RCV.2.2** For

- [assignment: *list of failures/service discontinuities*]

the TSF shall ensure the return of the TOE to a secure state according to the SSP using automated procedures.

128 Explanatory Note 25: An example for an instantiation of the list of failures may be "TOE initialization error", "TOE run-time error", "partition initialization error", "partition run-time error".

129 Explanatory Note 26: The SSP may be configured to a secure state for each kind of failure, for example, to halt the entire TOE, restart a partition or to ignore an error. Handling of TOE initialization errors according to the SSP is only possible after the TSF's mechanism to handle errors is active.

### *6.1.7 Class FRU Resource Utilization*

#### FRU_PRS.1 Limited Priority of Service

Hierarchical to: No other components.

Dependencies: No dependencies.

**FRU_PRS.1.1** The TSF shall assign a priority to each subject in the TSF.

**FRU_PRS.1.2** The TSF shall ensure that each access to **CPU resources**[66], [assignment: *further controlled resources*] shall be mediated on the basis of the subject's assigned priority.

#### FRU_RSA.2 Minimum and Maximum Quotas

130 **FRU_RSA.2/AS.USER_PART_RES for Asset: 'User Partition Resources'**

Hierarchical to: FRU_RSA.1.

Dependencies: No dependencies.

[66] [assignment: *controlled resources*]

**FRU_RSA.2.1** For each 'user partition', the TSF shall enforce maximum quotas of the following resources:

- **System memory: the maximum amount of physical memory that can be allocated to a partition;**
- **Processing time: each user partition is confined to the time window(s) as specified by the SSP[67]**
- [assignment: *further controlled resources*]

that *user applications executed in the corresponding partition*[68] can use *simultaneously*[69].

**FRU_RSA.2.2** For each 'user partition', the TSF shall ensure the provision of minimum quantity of each

- **System memory: the minimum amount of physical memory that can be allocated to the user partition;**
- **Processing time: each user partition gets access to its time window(s) within the partition schedule as specified in the SSP[70]**
- [assignment: *further controlled resources*]

that is available for *user applications executed in the corresponding partition*[71] to use *simultaneously*[72].

131 Explanatory Note 27: The CC text has been extended with "for each user partition" to indicate that resources shall be assigned per user partition.

### 132 FRU_RSA.2/AS.COMMUN_OBJ_RES for Asset: 'Communication Object Resources'

Hierarchical to: FRU_RSA.1

Dependencies: No dependencies.

**FRU_RSA.2.1** For each 'communication object', the TSF shall enforce maximum quotas of the following resources:

- **System memory: the maximum amount of physical**

[67] [assignment: *controlled resources*]
[68] [selection: *individual user, defined group of users, subjects*], refinement
[69] [selection: *simultaneously, over a specified period of time*]
[70] [assignment: *controlled resources*]
[71] [selection: *individual user, defined group of users, subjects*], refinement
[72] [selection: *simultaneously, over a specified period of time*]

**memory that can be allocated to the communication object;**[73]

- [assignment: *further controlled resources*]

that *user applications*[74] can use *simultaneously*[75].

**FRU_RSA.2.2** For each 'communication object', the TSF shall ensure the provision of minimum quantity of ~~each~~

- **System memory: the minimum amount of physical memory that can be allocated to a communication object;**[76]
- [assignment: *further controlled resources*]

that is available for *user applications and system applications*[77] to use *simultaneously*[78].

133 Explanatory Note 28: The CC text has been extended with "for each communication object" to indicate that resources shall be assigned per communication object.

### 134 FRU_RSA.2/AS.SYS_COMP_RES for Asset: 'System Component Resources'

Hierarchical to: FRU_RSA.1

Dependencies: No dependencies.

**FRU_RSA.2.1** For each 'system component', the TSF shall enforce maximum quotas of the following resources:

- [assignment: *controlled resources*]

that *system applications executed in the corresponding system component*[79] can use *simultaneously*[80].

**FRU_RSA.2.2** For each 'system component', the TSF shall ensure the provision of minimum quantity of each

- **System memory: the minimum amount of physical memory that can be allocated to the system**

[73] [assignment: *controlled resources*]
[74] [selection: *individual user, defined group of users, subjects*], refinement
[75] [selection: *simultaneously, over a specified period of time*]
[76] [assignment: *controlled resources*]
[77] [selection: *individual user, defined group of users, subjects*], refinement
[78] [selection: *simultaneously, over a specified period of time*]
[79] [selection: *individual user, defined group of users, subjects*], refinement
[80] [selection: *simultaneously, over a specified period of time*]

**component;**

- **Processing time: if the system component is a system partition it gets access to its time window(s) within the partition schedule as specified in the SSP**[81]
- [assignment: *further controlled resources*]

that is available for *system applications executed in the corresponding system component*[82] to use *simultaneously*[83].

135 Explanatory Note 29: The author of a ST may decide not to define any maximum quotas for system component resources in FRU_RSA.2.1.

136 Explanatory Note 30: The CC text has been extended with “for each system component” to indicate that resources shall be assigned per system component.

[81] [assignment: *controlled resources*]

[82] [selection: *individual user, defined group of users, subjects*], refinement

[83] [selection: *simultaneously, over a specified period of time*]

## 6.2 Security Assurance Requirements for the TOE

137 This PP claims conformance to the EAL5 augmented with AVA_VAN.5.

### *6.2.1 Security Requirements Rationale*

138 The following table provides an overview for security functional requirements coverage also giving an evidence for sufficiency and necessity of the SFRs chosen.

| | OT.AUDIT | OT.CONFIDENTIALITY | OT.INTEGRITY | OT.RESOURCE_AVAILABILITY | OT.SAFE_SECURE_STATE | OT.SYSTEM_APPLICATION_API_PROPTECTION |
|---|---|---|---|---|---|---|
| FAU_GEN.1 | X | | | | | |
| FDP_ACC.2/AS.USER_PART_CONT | | X | X | | | |
| FDP_ACC.2/AS.COMMUN_OBJ_CONT | | X | X | | | |
| FDP_ACC.2/AS.SYS_COMP_CONT | | X | X | | | |
| FDP_ACC.2/AS.AUD | X | X | X | | | |
| FDP_ACF.1/AS.USER_PART_CONT | | X | X | | | |
| FDP_ACF.1/AS.COMMUN_OBJ_CONT | | X | X | | | |
| FDP_ACF.1/AS.SYS_COMP_CONT | | X | X | | | |
| FDP_ACF.1/AS.AUD | X | X | X | | | |
| FDP_IFC.2 | | X | | | | |
| FDP_IFF.1 | | X | | | | |
| FDP_IFF.5 | | X | | | | |
| FDP_RIP.2 | | X | | | | |
| FIA_UID.2 | X | X | X | | | |

| | OT.AUDIT | OT.CONFIDENTIALITY | OT.INTEGRITY | OT.RESOURCE_AVAILABILITY | OT.SAFE_SECURE_STATE | OT.SYSTEM_APPLICATION_API_PROPTECTION |
|---|---|---|---|---|---|---|
| FMT_MOF.1 | X | | | | | |
| FMT_MSA.1 | X | X | X | | | |
| FMT_MSA.3 | X | X | X | | | |
| FMT_MTD.1/AS.USER_PART_RES | | X | X | | | |
| FMT_MTD.1/AS.USER_PART_SHAPE | | X | X | | | |
| FMT_MTD.1/AS.COMMUN_OBJ_RES | | X | X | | | |
| FMT_MTD.1/AS.COMMUN_OBJ_SHAPE | | X | X | | | |
| FMT_MTD.1/AS.SYS_COMP_RES | | X | X | | | |
| FMT_MTD.1/AS.SYS_COMP_SHAPE | | X | X | | | |
| FMT_MTD.1/AS.CONF_DATA | | X | X | | | |
| FMT_MTD.1/AS.SYS_APP_API | | | | | | X |
| FMT_SMF.1 | X | | | | | |
| FMT_SMR.1 | X | X | X | | | |
| FPT_FLS.1 | | | | | X | |
| FPT_RCV.2 | | | | | X | |
| FRU_PRS.1 | | | | X | | |
| FRU_RSA.2/AS.USER_PART_RES | | | | X | | |
| FRU_RSA.2/AS.COMMUN_OBJ_RES | | | | X | | |

| | OT.AUDIT | OT.CONFIDENTIALITY | OT.INTEGRITY | OT.RESOURCE_AVAILABILITY | OT.SAFE_SECURE_STATE | OT.SYSTEM_APPLICATION_API_PROPTECTION |
|---|---|---|---|---|---|---|
| FRU_RSA.2/AS.SYS_COMP_RES | | | | X | | |

Table 6: Coverage of Security Objectives for the TOE by SFR. “X” is for where a dependency to an objective exists.

### 139 Security Objective: OT.AUDIT

FMT_SMF.1 specifies a security management function on audit generation. FMT_MOF.1 controls usage of the security management function on audit generation. FAU_GEN.1 ensures that when the audit function is active the system collects audit data on events to be audited as defined by the SSP, including the date and time of the event, type of event, subject identity (if applicable), and the outcome (success or failure) of the event. FDP_ACC.2/AS.AUD, FDP_ACF.1/AS.AUD control that audit data can be *queried* by user applications and *treated* by system applications according to the SSP.

FIA_UID.2 ensures that applications are identified; FMT_SMR.1 provides security roles to applications; FMT_MSA.1 forbids modification of security attributes; and FMT_MSA.3 provides default values for security attributes.

### 140 Security Objective: OT.CONFIDENTIALITY

FDP_ACC.2/AS.USER_PART_CONT, FDP_ACF.1/AS.USER_PART_CONT, FDP_ACC.2/AS.COMMUN_OBJ_CONT, FDP_ACF.1/AS.COMMUN_OBJ_CONT, FDP_ACC.2/AS.SYS_COMP_CONT, FDP_ACF.1/AS.SYS_COMP_CONT, FDP_ACC.2/AS.AUD, FDP_ACF.1/AS.AUD ensure that all query accesses to user data as object are restricted to query accesses allowed according to the SSP.

FDP_IFF.1, FDP_IFC.2, FDP_IFF.5 ensure that information flows originating from user partitions are restricted to information flows allowed according to the SSP, ensuring separation of user partitions in space and time. FDP_RIP.2 ensures that no residual information is in CPU registers or memory caches according to the SSP, when CPU(s) are reused on a partition switch..

FMT_MTD.1/AS.USER_PART_RES, FMT_MTD.1/AS.USER_PART_SHAPE, FMT_MTD.1/AS.COMMUN_OBJ_RES**,** FMT_MTD.1/AS.COMMUN_OBJ_SHAPE, FMT_MTD.1/AS.SYS_COMP_RES, FMT_MTD.1/AS.SYS_COMP_SHAPE, FMT_MTD.1/AS.CONF_DATA ensure that all query accesses to TSF data as object are restricted to query accesses allowed according to the SSP.

FIA_UID.2 ensures that applications are identified; FMT_SMR.1 provides security roles to applications; FMT_MSA.1 forbids modification of security attributes; and FMT_MSA.3 provides default values for security attributes.

### 141 Security Objective: OT.INTEGRITY

FDP_ACC.2/AS.USER_PART_CONT, FDP_ACF.1/AS.USER_PART_CONT, FDP_ACC.2/AS.COMMUN_OBJ_CONT, FDP_ACF.1/AS.COMMUN_OBJ_CONT, FDP_ACC.2/AS.SYS_COMP_CONT, FDP_ACF.1/AS.SYS_COMP_CONT, FDP_ACC.2/AS.AUD, FDP_ACF.1/AS.AUD ensure that all modify accesses to user data as object are restricted to modify accesses allowed according to the SSP.

FMT_MTD.1/AS.USER_PART_RES, FMT_MTD.1/AS.USER_PART_SHAPE, FMT_MTD.1/AS.COMMUN_OBJ_RES**,** FMT_MTD.1/AS.COMMUN_OBJ_SHAPE, FMT_MTD.1/AS.SYS_COMP_RES, FMT_MTD.1/AS.SYS_COMP_SHAPE, FMT_MTD.1/AS.CONF_DATA ensure that all modify accesses to TSF data as object are restricted to modify accesses allowed according to the SSP.

FIA_UID.2 ensures that applications are identified; FMT_SMR.1 provides security roles to applications; FMT_MSA.1 forbids modification of security attributes; and FMT_MSA.3 provides default values for security attributes.

### 142 Security Objective: OT.RESOURCE_AVAILABILITY

FRU_RSA.2/AS.USER_PART_RES ensures that allocation limits are enforced on the minimum and maximum amount of memory and processing time available to a user partition.

FRU_RSA.2/AS.COMMUN_OBJ_RES ensures that allocation limits are enforced on the minimum and maximum amount of memory available to a communication object.

FRU_RSA.2/AS.SYS_COMP_RES ensures that allocation limits are enforced on the minimum amount of memory and, if applicable, processing time available to a system component.

If the SSP defines that subjects from different user partitions share the same time window, FRU_PRS.1 ensures priority-based CPU access.

143 **Security Objective: OT.SAFE_SECURE_STATE**

FPT_FLS.1 ensures the preservation of a secure state after failures. FPT_RCV.2 ensures that automated recovery from error conditions is possible. Initial reaching of the secure state is ensured by obligations on the operational environment defined in Section 3.3, organizational security policies P.SYSTEM_INTEGRATOR and P.SYSTEM_OPERATOR as well as the architectural property of secure initialization.

144 **Security Objective: OT.SYSTEM_APPLICATION_API_PROTECTION**

FMT_MTD.1/AS.SYS_APP_API ensures that the TOE prevents any invocation of the system application API by a user application.

### *6.2.2 Security Assurance Requirements Rationale*

EAL5+ has been considered appropriate to ensure the robust and reliable separation of partitions.

An operating system providing a generic MILS separation kernel needs to be at least as trustworthy as its guest applications, which also is an argument for a high degree of assurance.

A MILS separation kernel needs to communicate that it is NEAT (non-bypassable, evaluable, always-invoked and tamperproof [14]). Demonstrating NEAT properties is an important argument for performing vulnerability requirements along a high level of AVA_VAN.5. The selection of the component AVA_VAN.5 provides a higher assurance than the pre-defined EAL5 package, namely requiring a vulnerability analysis to assess the resistance to penetration attacks performed by an attacker possessing a high attack potential.

The whole architecture of the separation kernel shall be implemented in a modular way as required by EAL5 to allow easy and thorough inspection for the NEAT properties.

Explanatory Note 31: In particular, EAL 5 has also been identified as good match for high-criticality avionics products [12].

### *6.2.3 Security Assurance Requirements Dependency Analysis*

145 In this section, we provide a dependency analysis for the security assurance requirements as defined by the CC. There are no unfulfilled dependencies.

146 This PP claims conformance to the standard EAL5 package augmented with AVA_VAN.5. For the EAL5 standard package, all dependencies in CC v3.1 part 3 provided packages are fulfilled. In addition, this PP also provides a dependency analysis for the security assurance requirement AVA_VAN.5.

147 AVA_VAN.5 depends on: ADV_ARC.1: fulfilled by ADV_ARC.1; ADV_FSP.4 hierarchically fulfilled by ADV_FSP.5; ADV_IMP.1: fulfilled by ADV_IMP.1; ADV_TDS.3: hierarchically fulfilled by ADV_TDS.4; AGD_OPE.1: fulfilled by AGD_OPE.1; AGD_PRE.1: fulfilled by AGD_PRE.1; ATE_DPT.1: hierarchically fulfilled by ATE_DPT.3.

# Chapter 7 Acknowledgement

148 Part of this PP is based on SKPP [6] [7], OSPP [8], HASK-PP [5], the security functional group approach is from [13].

# Chapter 8 Glossary

**Application:** An *application* is executable data. It is either a system application or a user application.

**Attacker:** An attacker is a threat agent (a person or a process acting on his/her behalf) trying to undermine the TOE security policy defined by the current PP and, hence, the SSP. The attacker especially tries to change properties of the assets having to be maintained according to the TOE security policy defined by the current PP (see Table 1 and Table 2 in Section 3.1.1). The attacker is assumed to possess an at most high attack potential.

Note that the TOE security policy defined by the current PP only addresses attacks carried out by user applications and does not address any physical attacks.

**Audit Data:** *Audit data* is electronic records reflecting events to be audited.

**Bootloader:** A *bootloader* is software that loads the TOE on the hardware and hands over the full control to the TOE. In particular, a TOE-external check of the TOE may be implemented in the bootloader (e.g. for “secure boot”).

**Communication Object:** Partitions can communicate with each other under the supervision of the TOEs separation kernel. A *communication object* is an object exposed to one or multiple partitions with access rights as defined in the configuration data. The content of a communication object is the content of a communication object and exchanged (received and sent) between partitions. The resources of a communication object are physical memory space.

**Configuration Data:** *Configuration data* is data used by the TOE to enforce the SSP.

The configuration data defines a set of rules on how the TOE behaves. For example, the configuration data comprises the assignment of resources and communication objects to partitions. The configuration data is defined during Step 2 of the generic Lifecycle (Section 1.3.4.2).

The default configuration is that there is no information flow between any partitions. Any information flow between partitions has to be explicitly allowed by the system integrator in the configuration data.

**Content:** *Content* can be either the content of a user partition or a system partition or a communication object*.* The content of a user partition is user applications and/or data being executed and/or stored in a user partition. The content of a system component is system applications and/or data being executed and/or stored in the system component, supplied by the system integrator. The content of a communication object is the content of a communication object and exchanged (received and sent) between partitions.

**Events to be Audited:** The system integrator selects the *events to be audited*, that is the internal TOE events to be detected and recorded by the TOE.

**Firmware:** *Firmware* is software and data stored in non-volatile memory of the hardware that initializes the hardware after the power on.

**Hardware:** *Hardware* is the physical part of the TOE operational environment on which the TOE is executed. Usually, hardware is a board with several components such as CPUs, serial interfaces, network adapters, I/O devices etc. There are Separation Kernel Hardware Abstraction Layer controlled components (e.g. CPUs, caches) and ODSP controlled components (e.g. serial interfaces, timer). This PP considers the following parts as part of the hardware: bootloader, firmware.

**Separation Kernel Hardware Abstraction Layer:** A *Separation Kernel Hardware Abstraction* Layer (SK-HAL) provides specific low-level functionality for each supported CPU architecture. Since the CPU instruction set is also CPU dependent, the generic components are CPU specific at the object code level.

The usual responsibility of an SK-HAL may comprise: (1) abstraction of data type representation, (2) processor exception handling, and (3) low level address space and memory management.

In operational use, the TOE always contains only one SK-HAL.

**Instruction Set Architecture:** The *instruction set architecture* is the set of instructions available to operate on a CPU provided by a CPU manufacturer.

**Life Cycle:** The typical *life cycle* phases for this kind of TOE are development (source code development), manufacturing (compilation to binary), system integration (by the system integrator), installation (by the system operator), and finally, operational use (by the system operator). Operational use of the TOE is explicitly in the focus of this PP.

**Modify:** The verb "*modify"* is used to describe an addition to, change to or deletion of data; dependent on a concrete context, "modify" is to be considered as synonym for "write".

**Object:** An *object* is a passive entity in the TOE manipulated by subjects with operations. In policies, subjects are related to objects by authorizations. This defines the way objects may be accessed by subjects. Objects are listed in Section 3.1.1.

**Operational Policy for the Product in the Field:** The *operational policy for the product in the field* covers the life cycle phase "operational use". It is a set of rules issued by the system integrator how the product in the field is to be operated. The system integrator obliges the system operator to follow this policy.

**Partition:** A *partition* is a logical unit maintained by the separation kernel and configured by the SSP. A *partition* contains user data. For each partition, the separation kernel provides resources. Resources of a partition comprise physical memory space, I/O memory space, a description of the set of CPUs the partition's applications can run on, allocated CPU time for each CPU, and interrupts.

**Partitioned Information Flow Control Policy (PIFP):** The system integrator can derive a *Partitioned Information Flow Policy* (PIFP) from the SSP, applying the following rule: A PIFP information flow from a partition *A* to a partition *B* is allowed if and only if there exists a communication object in the SSP that *A* may modify to and *B* may query.

**Partition Isolation:** A partition switch occurs when a CPU(s) is/are assigned to another partition. Partition switches are defined by SSP as part of the scheduling scheme. The TSF enforces that no residual information is in CPU registers or memory caches according to the SSP.

**Partition Switch:** A partition switch occurs when a CPU(s) is/are assigned to another partition. Partition switches are defined by SSP as part of the scheduling scheme. The TSF enforces that no residual information is in CPU registers or memory caches according to the SSP.

**TOE Operating System:** The TOE *operating system* consists of the separation kernel and TSF data.

**TOE Separation Kernel:** The separation kernel provides the TSF and operates the TOE, by implementing mechanisms to assign resources to partitions, providing the execution environments for applications, and implementing communication between partitions as defined by the configuration data.

**On-board device Support Package:** An *on-board device support package* is a special purpose HAL and may contain a set of drivers for specific hardware components (a system application). It is supplied *and approved by the* system integrator. An on-board device support package can be exchanged without changing the separation kernel binary image, the content of any other partition or the content of *a system component of* the TOE. An *on-board device support* package uses the TSF's on-board device support package API. In operational *use, the TOE* always contains only one *on-board device support package*. The main tasks of a ODSP are (1) platform initialization, (2) interrupt management, (3) hardware timer management, (4) memory region management.

**Product Binary Image:** The *product binary image* is the output of the generic Lifecycle (Section 1.3.4.2). The product binary image contains the TOE separation kernel binary image, the configuration data in a representation readable by the product binary image, the content of the on-board device support package, the content of system extensions and the content of partitions. The system integrator provides this product binary image to the system operator who, at the system operator's site, installs it on the hardware. During operational use, user applications cannot change the product binary image, e.g. no new user or system partitions can be created, no new communication objects can be created, no new user or system applications can be loaded.

**Query:** The verb "*query*" is used to describe to extract data directly or to derive them from a representation (e.g. log data), based on specified conditions; dependent on a concrete context, "query" is to be considered as synonym for "read", "read out", and "execute".

**Resource:** In this PP we consider *resources* of partitions, communication objects and system components. The resources of a partition comprise physical memory space, I/O memory space, a description of the set of CPUs the partition's applications can run on, allocated CPU time for each CPU, and interrupts The resources of a communication object are physical memory space. The resources of a system component comprise physical memory space, I/O memory space, a description of the set of CPUs the system component's applications can run on, for system partitions, allocated CPU time for each CPU, and interrupts.

**Resource Usage Data:** *Resource usage data* is data accounting for the usage of resources. For example, the partition resource usage data accounts for how much memory a partition has already used and how much there is still available. Resource usage data is stored in shapes. The TSF protects the confidentiality and integrity of resources and shapes.

**Safe and Secure State:** A *safe and secure state* is a state in which the TOE enforces the SSP. The safe and secure state is maintained by a scheme for automatic handling of error conditions (configured in Step 2 of Section 1.3.4.2).

**Shape:** A *shape* is TSF data that contains an entity's identity, the entity's resource usage data, a set of security attributes according to the SSP assigned to the entity, and links the content assigned to an entity to the resources assigned to the entity.

**Subject:** A *subject* is an active entity that can perform operations on objects. A subject requires resources provided by the TOE to become operational. A subject is an abstraction created by the TSF. Subjects are listed in Section 3.1.2.

**System Application:** A *system application* is any application within a system partition, a system extension, or the on-board device support package (ODSP). Only a system application in a system partition is allowed to use the TOE system partition API. Only a system application in a system extension is allowed to use the TOE system extension API. Only a system application in the ODSP is allowed to use the TOE ODSP API.

**System Application API:** The *system application API* is an interface to functions of the TSF available for system applications. The system application API is the combined functionality of the PikeOS system partition API, the PikeOS system extension API, and the PikeOS ODSP API. Only a system application in a system partition is allowed to use the TOE system partition API. Only a system application in a system extension is allowed to use the TOE system extension API. Only a system application in the ODSP is allowed to use the TOE ODSP API.

**System Component:** A *system component* is a system partition (Section 1.3.2.2.2 above), system extension (Section 1.3.2.4 below), or an ODSP (Section 1.3.2.5 below). A system component contains user data supplied and approved by the system integrator.

**System Extension:** A *system extension* contains a software component (a system application) supplied and approved by the system integrator and coupled with the separation kernel via the system extension API. A system extension can provide specific functionality to applications within partitions only under supervision of the separation kernel. A system extension can be exchanged without changing the separation kernel binary image, the content of any other partition or the content of a system component of the TOE.

**System Integration Policy (SIP):** The *system integration policy* (SIP) is a set of rules issued by the system integrator for using and protecting assets. The SIP also defines events to be audited.

The SIP is defined during the generic Lifecycle (Section 1.3.4.2), which can be split into the three steps: selection of the TOE operational environment and system applications and user applications (Step 1), configuration of the TOE (Step 2), and integration (Step 3). The result of performing Step 1 and Step 2 is that a SIP has been defined.

**System Integrator:** A *system integrator* is a person trusted to (re-)configure and integrate the TOE. This includes identifying system partitions and user partitions and assigning applications into partitions.

**System Operator:** A *system operator* is a person trusted to (re-)install, stop, start, restart, or access (also physically) the TOE in the field.

**System Partition:** A *system partition* contains applications and/or data supplied and approved by the system integrator. An application in a system partition is a *system application* and uses the system partition API of the separation kernel. The content of a system partition can be exchanged without changing the separation kernel binary image, the content of any other partition or the content of a system component of the TOE.

**System Security Policy (SSP):** The *System Security Policy* (SSP) consists of configuration choices made by a system integrator based on the subset of the configuration data rules evaluated in this PP. The SSP is enforced by the TSF and it cannot be circumvented by malicious user applications.

**Time Window:** A *time window* is assigned CPU time a to user application. User applications hosted in different user partitions can be assigned to different time windows according to the SIP.

**TOE Security Service:** A *TOE Security Service* is a logical part of the TOE that has to be relied upon for enforcing a related subset of the rules regulating how the SSP is maintained by the TOE.

**TOE User Manuals:** The *TOE User Manuals* are documentation provided with the TOE on how to use the TOE in general environments and in safety and security critical environments.

**Treat:** The verb “*treat”* is used as a synonym for “query” and “modify”. It describes all possible operations by a subject on an asset.

**User:** A *user* is an external entity. External entities are listed in Section 3.1.2.

**User Application:** A *user application* is any application within a user partition. A user application is allowed to use only the TOE user partition API. User applications can even be malicious, and even in that case the TOE ensures that malicious user applications are neither harming the TOE nor other applications in other partitions.

**User Application Developer:** A *user application developer* is a developer of an application that has been placed into a user partition by the system integrator.

**User Partition:** A *user partition* is defined as such by system integrator by an appropriate definition of the SSP. The content of a user partition is user applications and/or data being executed and/or stored in a user partition. User data can be executable and/or non-executable. The organizational security policy **P.SYSTEM_INTEGRATOR** requires that into any user partition, the system integrator only loads user applications.

# Chapter 9 Abbreviations

API: Application Programming Interface

CC: Common Criteria for Information Technology Security Evaluation

CPU: Central Processing Unit

DMA: Direct Memory Access

EAL: Evaluation Assurance Level

HASK: High-Assurance Security Kernel

ISA: Instruction Set Architecture

I/O: Input / Output

IT: Information Technology

MILS: Multiple Independent Levels of Security

MMU: Memory Management Unit

NEAT: non-bypassable, evaluable, always-invoked and tamperproof

ODSP: On-board Device Support Package

OSP: Organizational Security Policy

OSPP: Operating Systems Protection Profile

PIFP: Partitioned Information Flow Policy

SAR: Security Assurance Requirement

SFG: Security Functional Requirement Group

SFP: Security Function Policy

SFR: Security Functional Requirement

SIP: System Integration Policy

SK-HAL: Separation Kernel Hardware Abstraction Layer

SKPP: Separation Kernel Protection Profile

SSP: System Security Policy

ST: Security Target

TOE: Target of Evaluation

TSF: Target of Evaluation Security Functionality

TSFI: TSF Interface

TSS: TOE Summary Specification

TSS_XXX: TOE Security Service XXX

# Chapter 10 Bibliography

**Common Criteria**

[1] Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model; Version 3.1, Revision 4, September 2012, CCMB-2012-09-001

[2] Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Components, Version 3.1, Revision 4, September 2012, CCMB-2012-09-002

[3] Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Requirements; Version 3.1, Revision 4, September 2012, CCMB-2012-09-003

[4] Common Methodology for Information Technology Security Evaluation, Evaluation Methodology; Version 3.1, Revision 4, September 2012, CCMB-2012-09-004

**Protection Profiles**

[5] Protection Profile for High-Assurance Security Kernel, Bundesamt für Sicherheit in der Informationstechnik (BSI) and Sirrix AG security technologies, Version 1.14, June 2008

[6] U.S. Government Protection Profile for Separation Kernels in Environments Requiring High Robustness, Information Assurance Directorate,. Version 1.03, June 2007

[7] Separation Kernel Protection Profile revisited: Choices and rationale, Timothy E. Levin, Thuy D. Nguyen, Cynthia E. Irvine, Michael McEvilley, 4th Annual Layered Assurance Workshop (LAW), 2010

[8] Operating System Protection Profile, Stephan Mueller, Gerald Krummeck, Helmut Kurth, 2010

**Other Sources**

[9] Design and verification of secure systems, 8th ACM Symposium on Operating System Principles, John Rushby, 1981

[10] The MILS architecture for a secure global information grid, W. Scott Harrison, Nadine Hanebutte, Paul Oman, Jim Alves-Foss, CrossTalk 18 (10), p. 20–24, 2005

[11] The MILS architecture for high assurance embedded systems, Jim Alves-Foss, W. Scott Harrison, Paul Oman, Carol Taylor, International Journal of Embedded Systems 2 (3/4), p. 239-247, 2006

[12] Towards Common Criteria certification for DO-178B compliant airborne software systems, Jim Alves-Foss, Bob Rinker, Carol Taylor, , 2002, http://www.csds.uidaho.edu/papers/Alves-Foss02b.pdf

[13] How to Create a slim and comprehensive PP: The Frame Approach, International Common Criteria Conferrence (ICCC), Igor Furgel, 2013, https://www.fbcinc.com/e/iccc/presentations/T3_D2_12pm_Furgel_How_to_create_a_slim_and_comprehensive_PP.pdf

[14] MILS virtualization for Integrated Modular Avionics, David Kleidermacher, Mike Wolf, 27th Digital Avionics Systems Conference, 2008

Secure European virtualisation for trustworthy applications in critical domains. The mission of the EURO-MILS project is to develop a solution for virtualization of heterogeneous resources and provide strong guarantee for isolation of resources by means of Common Criteria certification with usage of formal methods.

**www.euromils.eu**

**for further information please contact the coordinator**

**TECHNIKON Forschungs- und Planungsgesellschaft mbH**

**coordination@euromils.eu**