D2.1
Base MILS Platform Protection Profile

Project number:	731456
Project acronym:	certMILS
Project title:	Compositional security certification for medium to high-assurance COTS-based systems in environments with emerging threats
Start date of the project:	1st January, 2017
Duration:	48 months
Programme:	H2020-DS-LEIT-2016

Deliverable type:	Report
Deliverable reference number:	DS-01-731456/ D2.1/ 1.0
Work package contributing to the deliverable:	WP 2
Due date:	April 2018 (M16)
Actual submission date:	4th May, 2018

Responsible organisation:	Epoche and Espri
Editor:	Jose Emilio Rico
Dissemination level:	PU
Revision:	1.0

Abstract:	This document is the Base MILS Platform Protection Profile (PP). PPs, according to the Common Criteria, represent templates for Security Targets and specify security functions and the security environment of classes of products. This PP serves as a "base PP" describing generic properties of Separation Kernels. Optional functionality is covered by a set of PP modules that build on this PP.
Keywords:	Protection Profile, MILS Platform, Separation Kernel, PP Module, Base PP

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731456.

Editor

Jose Emilio Rico (E&E)

Contributors (ordered according to beneficiary numbers)

Helmut Kurth (ATSEC)

Andreas Hohenegger (ATSEC)

Jose Emilio Rico (E&E)

Alvaro Ortega (E&E)

Holger Blasum (SYSGO)

Sergey Tverdyshev (SYSGO)

Michal Hager (EZU)

Executive Summary

Whereas a Security Target (ST) always describes a specific Target of Evaluation (TOE), a Protection Profile (PP) is intended to describe a TOE type - in this case a Separation Kernel (SK) operating system. The same PP may therefore be used as a template for many different STs employed in different Common Criteria evaluations.

This “Base PP” can be extended using defined “PP Modules” that are produced as certMILS deliverable D2.2. These PP modules describe optional security functionality of an SK (that may cover additional threats). Together with the base PP, these build a “PP Configuration”.

The base PP first provides an overview of the type of TOE. The relevant normative text is identified (Common Criteria version 3.1 revision 5) and the security problem is defined. The resolution of that security problem is achieved through security objectives for the TOE and its environment. These objectives are re-expressed as formal security functional requirements that the TOE must implement and as assumptions that the environment must satisfy.

A PP is a living document, so this deliverable should be perceived as a snapshot of the certMILS partners at M16 that will be constantly extended, based on the internal and external feedback.

Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Terms and Definitions

Authorised entity/authorised user: As in CC Part 2, the term authorised user (and analogously an authorised entity) is used to signify a user (entity) who/which possesses the rights and/or privileges necessary to perform an operation. Used in SFRs, the term, therefore, indicates that it is allowable for a user to perform a specific operation or a set of operations as defined by those SFRs. This PP (and associated PP modules) does not work out all of the operations as needed for a concrete instantiation of an SFR and, therefore, leaves these definitions to an ST author who needs to make them in accordance with the SSP.

Bootloader/Firmware: A Bootloader or Firmware is hardware-specific software which comprises the following:

- Software and data stored in non-volatile memory of the hardware that initializes the hardware after the power on.
- Software that (fully or partially) loads the TOE into RAM memory 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”).

Executable and Linkable Format (ELF) / Common Object File Format (COFF): Elf is a common file format used on a variety of systems for executable code such as applications. Similarly, COFF is a file format used on some (e.g. Unix based) systems for executable code.

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, memory and I/O devices (e.g. serial interfaces, network adapters) etc.

Memory Management Unit (MMU) / Memory Protection Unit (MPU): An MMU is a part of the hardware, usually integrated in the CPU, which is capable of restricting accesses (e.g. destinations of load and store CPU instructions) of non-privileged executables to certain memory regions. The MMU shall only be configurable from a privileged CPU mode, thus, it can only be configurable through the TOE to configure the policies specifying these access restrictions. These policies are part of the SSP. During TOE run time, these policies are represented as page tables used by the MMU. An MPU is a hardware part, alternative to an MMU, that also provides memory protection but may not have other features of an MMU.

Memory Page: A memory page is an aligned and contiguous area of memory of a CPU architecture dependent size (e.g. 4096 bytes).

Multiple Independent Levels of Security (MILS): This is a concept that seeks to establish the security of a system’s multiple zones of different criticality by requiring the implementation of a number of principles. The key component of such system is usually its platform, which may be a SK as targeted by this Protection Profile.

Partition: A partition is a logical unit maintained by the separation kernel and configured by the SSP. For each partition, the separation kernel provides resources. Resources of a partition comprise physical memory and allocated CPU time for each CPU.

System Security Policy (SSP): The set of configuration choices made to influence the behaviour of the TOE with respect to its security functions. To solve the security problem the part of the SSP affecting the SFRs is to be enforced by the TOE Security Function (TSF) such that it cannot be bypassed or tampered with by attackers. The SSP is implicit in this Protection Profile which leaves the configuration choices to the role in charge of making these decisions when the TOE is used in a product.

Time-Slicing: The splitting of available CPU usage time in time windows (or time slices) assigned to different partitions in order to share this resource. There are different techniques to achieve time-slicing.

1.2 Base PP Reference

Title: Base MILS Platform Protection Profile
Sponsor: certMILS Consortium
CC Version: 3.1 (Revision 5)
Assurance Level: EAL4+ALC_FLR.2
Version: 1.0
Keywords: Base PP, PP Module, Operating System, Separation Kernel, MILS

1.3 TOE Overview

The scope of this Protection Profile (PP) is a Separation Kernel (SK) that can serve as basis for a MILS platform. SKs are expected to be non-bypassable, evaluable (minimal trusted computing base), always invoked and tamperproof. Their task is to provide multiple containers, called “partitions” and to assign to each of them a set of resources. The minimum set of resources that it must be able to guarantee are computer memory and processing time. As such, the SK is an operating system with minimized functionality, typically leaving the implementation of higher level functions, usually provided by an operating system (like file systems, network protocols and application management) to the partitions.

While the basic functions of such SK are minimal, this PP is accompanied by a number of PP modules that address areas where common SKs handle functions in a different way. In particular, there are *mandatory* PP modules to address different techniques by means of which the TOE may share CPU usage time between different partitions (CPU time modules).

1.3.1 TOE Type

The targeted TOE is a special kind of operating system, namely a Separation Kernel (SK).

1.3.2 Usage and Major Security Features of the TOE

A Separation Kernel (SK) is a special kind of operating system that allows to effectively separate different partitions from each other. Applications themselves are hosted in those partitions. They can also be entire operating systems. The SK is installed and runs on a hardware platform (e.g. embedded systems, desktop class hardware).

SKs, as addressed by this PP, can be used as a basis for systems that need to isolate different applications executing on the same platform from mutual influence. The SK may only allow *controlled* flow of information between them. Alternatively, it may not allow any information flow at all. Use cases include, but are not limited to

- control systems in planes, cars, trains, space or production facilities,
- environments that require virtualization with strong separation between virtual machines (VMs),
- embedded systems that can host multiple applications which need to be separated from each other,
- high-assurance information gateways/re-grader/firewalls/guard systems/routers,
- mobile devices with critical functions that need to be separated from the general applications (e.g. user authentication, payment, file encryption),

- systems with requirements on modular/composable assurance.

An SK executes in the privileged mode of the underlying processor. Therefore, it can control the resources provided by the platform and restrict the access of partitions running in a less-privileged mode. At a minimum, it must be able to confine the partitions, and also control all communication channels. Further, it must ensure a minimum amount of each critical resource for a partition, that is, at least a well-defined or minimum part of main memory and a controlled access to CPU time.

Generic security features implemented by an SK are:

- separation in space: access control to memory assigned to partitions,
- separation in time: control of CPU time assigned to partitions,
- control of communication between partitions,
- non-bypassable, evaluable (minimal trusted computing base), always invoked, tamperproof.

The set of configuration elements that define the detailed behaviour of these and other security features, e.g. if and how partitions can communicate, is referred to as the System Security Policy (SSP).

1.3.2.1 Use as Small OS / Security Kernel

An SK can be used simply as a small-size operating system (OS). Its small code base ensures that the entire OS code can be verified and validated rigorously. Such OS is also called “microkernel”, “microvisor”, or “security kernel”. In this scenario, even a system with only one partition can be useful.

1.3.2.2 Static Mixed-Criticality Systems

A typical use case for an SK is that of mixed-criticality systems. The configuration of such a system assigns applications of different criticality to different partitions. Mixed-criticality systems also can host applications of equal criticality level if there is a requirement that these applications can interact only in a way that is controlled by the SK. Of course, the SK has to be certifiable according to the highest of the criticality levels of the applications in place.

1.3.2.3 Mixed-Criticality Systems with Secure Update

In these use cases, one or several applications, or the SK itself (including its configuration), can be updated. In case of an update of the applications of a single partition, the content of the partition is updated without affecting other partitions/applications. In case of an SK update, the partitioning mechanism enables and controls the update process, e.g. by implementing staged update / defense in depth. Appropriate authentication and authorization is either implemented in an application or in the SK itself.

1.3.2.4 OS for Dedicated Security Components

An SK can act as OS for systems with dedicated security components (DSC). DSCs are a composition of discrete hardware component domains and the inter-domain communication dedicated to the provisioning, protection, and use of the credentials employed by smart devices [DSC].

1.3.2.5 Secure Use of New Functionality or Legacy Software

An SK can be used to encapsulate new functionality, which shall not interfere with an existing, potentially already safety or security-certified system, into partitions. This additional functionality could be, for instance, a monitoring function, when an existing device is connected to the Internet.

In this case, the SK provides the basis for a virtualization system where an adapted general-purpose OS can execute within a partition and provide the same functions to its applications as the same OS executing directly on a hardware platform.

1.3.2.6 Systems with Trusted Partitions

Some SK implementations allow trusted privileged partitions. These may permit their host applications to partially or fully circumvent/update the SSP, e.g. to change the time-scheduling of partitions or to reassign memory. This kind of setup can also be useful to implement custom monitoring and control functionality. The SK shall still control the untrusted partitions according to the SSP in place.

1.3.3 Hardware/Software/Firmware not included in the TOE

The TOE consists of the SK software as well as, potentially, one or more trusted partitions used to configure the TOE and/or to perform TOE management functions etc. Excluded from the TOE are:

- the hardware of the underlying platform,
- the firmware/software that is specific to the underlying platform to initialize platform components/devices other than the processor and to provide the basic device drivers used to access devices that are part of the platform,
- trusted partitions that are not required for the operation of the TOE but implement functions that require privileges that the TOE provides.

Application Note: To fulfill its task, the SK may need specific functions of the platform such as the presence of an MMU in order to control the access to memory. An ST author claiming conformance to this PP shall name supported platforms and state any particular requirements of the SK on the platform.

1.4 TOE Description

1.4.1 TOE Architecture

The TOE covered by this PP consists of a kernel that operates in the highest privilege mode of the underlying processor and, potentially, one or more trusted partitions that support the kernel in its tasks. Such a trusted partition may be used, for example, to examine the configuration of partitions, instruct the kernel how to define the partitions and assign the resources to them.

Optionally, the TOE may also include one or more untrusted partitions, which are similar to untrusted applications that are delivered as a part of an operating system. The TSF of the TOE consists of the kernel and the trusted partitions.

As part of the TOE initialization, the TSF will obtain the information on the partitions to create and start the resources assigned to those partitions.

The initial number of partitions and the initial assignment of resources to a partition are configured statically and provided as input when the SK is started. Dynamic creation and reconfiguration of partitions are not within the scope of this Base PP.

Application Note: Such features are intended to be covered by optional PP modules.

The SK, in general, allows controlled communication between partitions. This is done by assigning communication ports to partitions that mediate the unidirectional or bidirectional exchange of data. Therefore, it can be configured to allow such interaction between identified partitions.

Application Note: Inter-partition communication is intended to be covered by optional PP modules.

The SK executes on a dedicated platform providing memory, processor(s), I/O devices and, possibly, other resources. The platform is considered as a part of the TOE environment as the hardware and basic firmware/software is concerned. Therefore, this PP does not make any assumptions on the underlying platform, except for the support to load the TOE software and the software executing within the individual partitions. It is assumed that the platform passes control to the TOE after it has performed its own initialization and that it provides the necessary interfaces the TOE needs to load additional software either for itself or for the partitions.

The SK may manage a set of privileges that it can assign to partitions, and thereby allow creating partitions that use functions associated with those privileges.

Application Note: Such privilege models, and how they are managed, are described by an optional PP module.

The SK may allow multiple active subjects per partition. Their number is either fixed (i.e. created when the partition is started) or dynamic, allowing a partition to create additional active subjects.

Application Note: There are different models for such a dynamic creation of new subjects and, therefore, this topic is also addressed by an optional PP module.

The SK provides interfaces to partitions, allowing them to request services from it. Those may be general programming interfaces like system calls, events that the SK intercepts from a partition, or call-back functions. In latter case, a partition instructs the SK to pass control to a dedicated entry point within the partition when the kernel detects a specific event that is related to the partition. Examples are exceptions caused by the partition or external interrupts that the SK associates with a specific partition.

This PP has no requirements for any of those interfaces except that their use shall not conflict with the SSP, does not allow a partition to elevate its privileges in an uncontrolled way, and does not tamper with the SK itself.

Note that an SK may allow for trusted partitions that have the potential to violate some of the requirements stated above. Such trusted partitions then need to be analysed to ensure that they do not misuse their privileges and have no side effects that could allow untrusted partitions to violate the policy enforced by the SK. A possible method to provide the assurance that trusted partitions adhere to the properties stated above is a compositional evaluation of the SK and such a trusted partition.

Figure 1 shows the generic architecture of the SK.

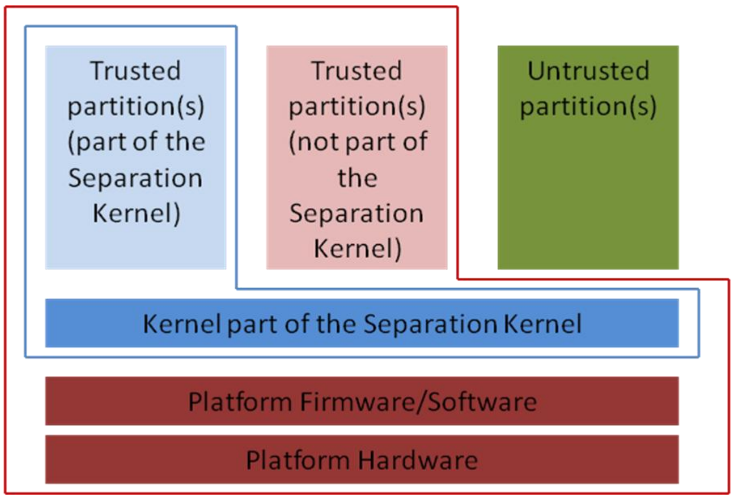
Trusted partition(s) (part of the Separation Kernel)
Trusted partition(s) (not part of the Separation Kernel)
Untrusted partition(s)
Kernel part of the Separation Kernel
Platform Firmware/Software
Platform Hardware

Figure 1: SK generic architecture.

The dark red components are mandatory parts of the TOE environment, the dark blue component is the mandatory part of the TOE, the light blue component is an optional part of the TOE architecture (there may be more than one trusted partition that is part of the TOE), the light red component is an optional part of the TOE environment. Note that the red components (both dark and light) need to be trusted in order for the SK to operate and enforce its policy.

In Figure 1, the blue line defines the TSF boundary for the SK, while the red line indicates the boundary for the trusted elements of a system that uses the TOE. The scope of this PP is the interior of the TSF boundary. The trust required for the elements within the red boundary outside of the TSF boundary can be established in separate evaluations that need to be composed with the evaluation of the SK.

1.4.2 CPU Usage Time Allocation

An SK may use different methods to allocate CPU time to partitions running on top of it. Such a time scheduling scheme ensures that each partition gets its turn and that the separation in time is guaranteed. The issue is particular sensitive for real-time operating systems (RTOS) which is subject to the additional requirement of processing data as it comes in and thus needs to provide a sufficient amount of CPU time to all real-time partitions.

In a single-core system, CPU usage time is wall-clock time. In a multicore system, the CPU usage time T is two-dimensional, it is the product of the number n of CPU cores and the wall-clock time t, i.e. $T = t * n$. In these systems T can be allocated according to the three different techniques listed below.

In view of the different options for time scheduling, the requirements of systems of either kind have been described in separate PP modules, each covering one of the following three basic categories:

- period-based scheduling [TM1],
- priority-based scheduling [TM2],
- CPU-affinity-based allocation (also referred to as CPU pinning) [TM3].

In systems using pure period-based time scheduling a time period is split into time windows. On each of these windows a fixed partition is scheduled. This time period, with the windows defined on it, is repeated invariably in a cycle.

In systems using pure priority-based time scheduling the partitions are assigned CPU time according to some kind of priority identified with each of them.

Finally, pure CPU-affinity-based allocation invariably assigns CPU cores to the partitions running on the SK.

In general, the time scheduling is assumed to be a combination of these three basic methods.

Application Note: An ST author claiming conformance to this PP must include at least one of these modules, but may also choose to include two or three of them if the time scheduling used by the TOE can only be represented as a combination of the three basic techniques.

Chapter 2 Conformance Claim

This PP claims conformance to

- Common Criteria for Information Technology Security Evaluation. Part 1: Introduction and general model. Version 3.1, Revision 5. April 2017. CCMB-2017-04-001 [CC1]
- Common Criteria for Information Technology Security Evaluation. Part 2: Security Functional Components. Version 3.1, Revision 5. April 2017. CCMB-2017-04-002 [CC2]
- Common Criteria for Information Technology Security Evaluation. Part 3: Security Assurance Components. Version 3.1, Revision 5. April 2017. CCMB-2017-04-003 [CC3]

as follows

- Part 2 conformant,
- Part 3 conformant.

The "Common Methodology for Information Technology Security Evaluation, Evaluation Methodology; Version 3.1, Version 3.1, Revision 5, April 2017. CCMB-2017-04-004, [CC4]" has to be taken into account.

2.1 PP Claim

This PP does not claim any conformance to other Protection Profiles.

2.2 Package Claim

This PP is conformant to the assurance package EAL4+ALC_FLR.2.

2.3 Conformance Rationale

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

2.4 Conformance Statement

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

Chapter 3 Security Problem Definition

This section describes the security aspects of the environment in which the TOE, claiming conformance with the PP, will be used and the manner in which the TOE is expected to be employed. It provides the statement of the TOE security environment, which identifies and explains all known and presumed threats countered by, either, the TOE or by the security environment, as well as assumptions about the secure usage of the TOE.

3.1 Assets

The basic assets (resources) an SK has to protect are described in Table 1.

Asset Name	Description	Security Properties to be Preserved
Memory (AS.MEM)	This may be just physical memory or physical and virtual memory. Memory may only be shared between partitions when this is deliberately configured. Application Note: An ST author shall specify if the SSP requires that any memory that is re-assigned during operation from one partition to another partition for exclusive use must be cleared before it can be accessed by the new partition.	Confidentiality, Integrity, Availability
CPU time (AS.TIME)	Processing time on a CPU or CPU core. In a single-core system, CPU usage time is wall-clock time. In a multicore system, the CPU usage time *T* is two-dimensional, it is the product of the number *n* of CPU cores and the wall-clock time *t*, i.e. $T = t * n$. Application Note: The SK may assign CPU cores for the sole use of a single partition or may use time-slicing (by period-based scheduling and/or priority-based scheduling) to allow that multiple partitions share all or some dedicated CPU cores. If CPU cores are shared, the algorithm to assign time-slices to partitions needs to ensure that – when required by the SSP – a partition receives a defined amount of CPU time within a defined time period.	Availability

Table 1: Assets and their security properties.

Application Note: Each ST that claims conformance to this PP needs to define assets not listed in this section and to state which of their security properties (confidentiality and/or integrity and/or availability) are to be preserved.

3.2 Threats

Threat agents are active subjects within an untrusted partition.

Application Note: Such an active subject may consist of any executable machine instructions that are loaded during start-up or runtime of the SK (i.e. become active) in the context of an untrusted partition. Executable machine instructions can come from, for example: (1) applications (e.g. binaries or libraries in formats such as ELF, COFF etc.) that the system integrator has initially installed, (2) on-the-fly downloads (including, e.g. “malware”) (3) on-the-fly compilation of source code.

The following threats apply to all assets with the respective security property.

T.DISCLOSURE

An attacker reads an asset for which the property confidentiality shall be preserved.

T.MODIFICATION

An attacker writes an asset for which the property integrity shall be preserved.

T.DEPLETION

By consuming resources for which the property availability shall be preserved, an attacker makes these resources unavailable to the TOE itself and/or to executables in untrusted partitions and/or to executables in trusted partitions.

Application Note: The assets concerned by these threats are identified via their security properties given in Table 1 and the corresponding listings in included PP modules.

3.3 Organizational Security Policies

There are no Organizational Security Policies for this PP.

3.4 Assumptions

A.TRUSTED_PARTITIONS

All trusted partitions are approved by an authorized person before being executed on the TOE. This person, thereby, takes responsibility that the trusted partitions have been developed according to the guidelines for trusted partitions provided by the TOE developer. This person, in particular, takes responsibility not to put untrusted executables into trusted partitions.

A.HARDWARE

The underlying hardware, firmware and bootloader needed by the TOE to guarantee secure operation provide the necessary properties, are working correctly and have no undocumented or unintended security critical side effect on the functions of the TOE.

A.EXCLUSIVE_RESOURCES

All resources required by the TOE, its trusted partitions, and all its other partitions are exclusively controlled by the TSF and not by any other entity outside of the TOE. The trusted components listed in A.HARDWARE merely assist the TOE in exerting this control.

A.PHYSICAL

The IT environment provides the TOE with appropriate physical security, commensurate with the value of the IT assets protected by the TOE.

A.TRUSTWORTHY_PERSONNEL

The personnel developing a product integrating the TOE and those configuring, installing and operating the TOE are trustworthy, act according to the guidance documentation and are sufficiently qualified for this task.

Chapter 4 Security Objectives

Security objectives are concise, abstract statements of the intended solution to the problem posed in the security problem definition (see Chapter 3). The set of security objectives for a TOE form a high-level solution to this security problem. It is divided into two part-wise solutions: the security objectives for the TOE, and the security objectives for the TOE's operational environment.

The subsequent sections present the solution to the security problem in terms of objectives of the two kinds.

4.1 Security Objectives for the TOE

OT.CONFIDENTIALITY

For each asset that requires confidentiality protection according to the definition of the assets, the TOE shall preserve its confidentiality.

OT.INTEGRITY

For each asset that requires integrity protection according to the definition of the assets, the TOE shall preserve its integrity.

OT.AVAILABILITY

For resources assigned to partitions the TOE shall preserve their availability.

Application Note: The assets concerned by these security objectives for the TOE are identified via their security properties given in Table 1 and the corresponding listings in included PP modules.

4.2 Security Objectives for the Operational Environment

OE.TRUSTED_PARTITIONS

All trusted partitions are approved by an authorized person before being executed on the TOE. This person, thereby, takes responsibility that the trusted partitions have been developed according to the guidelines for trusted partitions provided by the TOE developer. This person, in particular, takes responsibility not to put untrusted executables in trusted partitions.

OE.HARDWARE

The underlying hardware, firmware and bootloader needed by the TOE to guarantee secure operation provide the necessary properties, are working correctly and have no undocumented security critical side effect on the functions of the TOE.

OE.EXCLUSIVE_RESOURCES

All resources required by the TOE, its trusted partitions, and all its other partitions are exclusively controlled by the TSF and not by any other entity outside of the TOE. The trusted components listed in A.HARDWARE merely assist the TOE in exerting this control.

OE.PHYSICAL

The IT environment provides the TOE with appropriate physical security, commensurate with the value of the IT assets protected by the TOE.

OE.TRUSTWORTHY_PERSONNEL

The personnel developing a product integrating the TOE and those configuring, installing and operating the TOE are trustworthy, act according to the guidance documentation and are sufficiently qualified for this task.

4.3 Security Objectives Rationale

4.3.1 Coverage

Table 2 shows the one-by-one coverage of the threats and assumptions by the security objectives for the TOE and the operational environment, respectively.

	OT.CONFIDENTIALITY	OT.INTEGRITY	OT.AVAILABILITY	OE.TRUSTED_PARTITIONS	OE.HARDWARE	OE.EXCLUSIVE_RESOURCES	OE.PHYSICAL	OE.TRUSTWORTHY_PERSONNEL
T.DISCLOSURE	X							
T.MODIFICATION		X						
T.DEPLETION			X					
A.TRUSTED_PARTITIONS				X				
A.HARDWARE					X			
A.EXCLUSIVE_RESOURCES						X		
A.PHYSICAL							X	
A.TRUSTWORTHY_PERSONNEL								X

Table 2: Coverage of the Security Objectives.

4.3.2 Sufficiency Rationale

T.DISCLOSURE is countered directly by **OT.CONFIDENTIALITY** as it requires the TOE to protect assets whose confidentiality shall be preserved against unauthorized read accesses.

T.MODIFICATION is countered directly by **OT.INTEGRITY** as it requires the TOE to protect assets whose integrity shall be preserved against unauthorized modification accesses.

T.DEPLETION is countered directly by **OT.AVAILABILITY** as it requires the TOE to protect the assets whose availability shall be preserved accordingly.

A.TRUSTED_PARTITIONS is upheld directly by **OE.TRUSTED_PARTITIONS**.

A.HARDWARE is upheld directly by **OE.HARDWARE**.

A.EXCLUSIVE_RESOURCES is upheld directly by **OE.EXCLUSIVE_RESOURCES**.

A.PHYSICAL is upheld directly by **OE.PHYSICAL**.

A.TRUSTWORTHY_PERSONNEL is upheld directly by **OE.TRUSTWORTHY_PERSONNEL**.

Chapter 5 Extended Components Definition

There are no extended components in this PP.

Chapter 6 Security Requirements

The Security Functional Requirements (SFRs) provide a model of the security functionality of the TOE in semi-formal language. Likewise, the Security Assurance Requirements (SARs) describe the actions to be performed during the evaluation in order to gain assurance that the TOE works as claimed by an ST based on this PP and included modules.

This chapter defines the SFRs and the SARs and gives a rationale and dependency analysis for each.

6.1 Security Functional Requirements

In this PP and associated PP modules assignment, selection, and refinement operations are performed on the SFRs chosen from CC Part 2. The assignment operation is marked by square brackets "[*]", where the asterisk denotes the assignment. The selection operation is marked in italic. In a refinement operation added text is underlined and removed text is crossed out. The iteration operation is used when a component is repeated on varying assets (in this PP and in PP modules). Iteration is denoted by showing a slash ("/") and the iteration indicator after the component identifier. Assignments and selections that are to be performed by the PP user are highlighted as "[assignment: *]" and "[selection: *]" respectively. Here the asterisk represents the assignment to be made or the elements to select from.

Application Note: The SFRs in this base PP and in PP modules may define Security Functional Policies (SFPs), e.g. the "memory access control policy" introduced below. These SFPs are subsets of the SSP to be defined by the role configuring the TOE. The SSP may also define security attributes (subjects, objects and rules) used by an ST author to perform operations that are left open in the SFRs stated in the base PP and PP modules. An ST author may use this to describe a security functionality, which simultaneously allows many configurations, or even classes of configurations, of the TOE as usually required for an OS. The ST author shall ensure that the role configuring the TOE can take care that these configurations are secure. For instance, the ST author can point to appropriate user manuals by the manufacturer. That is, combined with the SSP, the SFRs stated in an ST claiming conformance to this base PP and PP modules must solve the technical security problem of its PP configuration.

6.1.1 User Data Protection

6.1.1.1 FDP_ACC.2/MA Complete Access Control – Memory Access

FDP_ACC.2.1/MA The TSF shall enforce the [memory access control policy] on [subjects: partitions, objects: memory areas] and all operations among subjects and objects covered by the SFP.

FDP_ACC.2.2/MA 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.

Application Note: Usually this function is implemented by the TSF using the MMU of the underlying CPU to set up (1) page tables, (2) memory protection attributes where supported by the CPU and (3) to map the memory into the address space of a partition to enforce this policy.

6.1.1.2 FDP_ACF.1/MA Security Attribute Based Access Control – Memory Access

FDP_ACF.1.1/MA The TSF shall enforce the [memory access control policy] to objects based on the following [subjects: partitions, objects: memory areas, [assignment: for partitions and memory areas, the SFP-relevant security attributes, or named groups of SFP-relevant security attributes]].

FDP_ACF.1.2/MA The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [assignment: set of rules that are used by the TSF to determine if access of a partition to a memory area is allowed].

Application Note: The rules to be assigned are the ones used by the TSF to define how memory is mapped into a partition's address space. Enforcement of those rules will use support of the underlying CPU which will raise an exception if a subject within a partition attempts to access memory that it is not allowed to access it at all or not allowed to access it in the requested mode.

FDP_ACF.1.3/MA The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: [assignment: rules, based on security attributes, that explicitly authorise access of subjects to memory].

FDP_ACF.1.4/MA 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 memory].

6.1.2 Identification and Authentication

6.1.2.1 FIA_UID.1 User Identification

FIA_UID.1.1 The TSF shall allow [assignment: list of TSF-mediated actions] on behalf of the ~~user~~ partition to be performed before the ~~user~~ partition is identified.

FIA_UID.1.2 The TSF shall require each ~~user~~ partition to be successfully identified before allowing any other TSF-mediated actions on behalf of that ~~user~~ partition.

6.1.3 Security Management

6.1.3.1 FMT_MSA.1/MA Management of Security Attributes – Memory Access

FMT_MSA.1.1/MA The TSF shall enforce the [memory access control policy] to restrict the ability to [selection: change_default, query, modify, delete, [assignment: other operations]] the security attributes [assignment: list of security attributes] to [assignment: the authorised identified entities].

6.1.3.2 FMT_MSA.3/MA Static Attribute Initialisation – Memory Access

FMT_MSA.3.1/MA The TSF shall enforce the [memory access control policy] 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/MA The TSF shall allow the [assignment: the authorised identified entities] to specify alternative initial values to override the default values when an object or information is created.

6.1.3.3 FMT_SMF.1 Specification of Management Functions

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].

6.1.4 Other Security Functional Requirements

To complement this PP, one or more of the CPU time modules have to be included. In particular, the SFRs contained in the selected modules are mandatory and address the security objectives that apply for the asset AS.TIME (compare Section 1.4.2).

6.2 Security Requirements Rationale

6.2.1 Coverage

Table 3 shows the coverage of the security objectives for the TOE by the SFRs.

	FDP_ACC.2/MA	FDP_ACF.1/MA	FMT_MSA.1/MA	FMT_MSA.3/MA	FIA_UID.1	FMT_SMF.1
OT.CONFIDENTIALITY	X	X	X	X	X	X
OT.INTEGRITY	X	X	X	X	X	X
OT.AVAILABILITY	X	X			X	

Table 3: Tracing of the security objectives for the TOE to the SFRs.

6.2.2 Sufficiency Rationale

Table 4 shows the sufficiency of the SFRs to address the security objectives for the TOE.

Security Objective	SFR
OT.CONFIDENTIALITY	The "memory access control policy" defined in **FDP_ACC.2/MA** and **FDP_ACF.1/MA** makes the confidentiality-protected asset of type AS.MEM not read-accessible to non-authorized entities. The SFRs **FMT_MSA.1/MA** and **FMT_MSA.3/MA** contribute in fulfilling this objective by providing management functions for the attributes associated to the "memory access control policy". **FIA_UID.1** ensures that partitions identify themselves against the TOE before TSF-actions may be performed and, thereby, provides the basis for the access control to memory and the CPU usage time allocation described in PP modules. **FMT_SMF.1** provides management functionality needed to ensure that the access to assets is properly controlled for read-access.
OT.INTEGRITY	The "memory access control policy" defined in

Security Objective	SFR
	FDP_ACC.2/MA and **FDP_ACF.1/MA** makes the integrity-protected assets contained in memory not write-accessible to non-authorized entities. The SFRs **FMT_MSA.1/MA** and **FMT_MSA.3/MA** contribute in fulfilling this objective by providing management functions for the attributes associated to the “memory access control policy”. **FIA_UID.1** ensures that partitions identify themselves against the TOE before TSF-actions may be performed and, thereby, provides the basis for the access control to memory and the CPU usage time allocation described in PP modules. **FMT_SMF.1** provides management functionality needed to ensure that the access to assets is properly controlled for write-access.
OT.AVAILABILITY	The access to the resources is controlled by **FDP_ACC.2/MA** and **FDP_ACF.1/MA** Provided that the SSP is defined properly, no uncontrolled/untrusted/unauthorized access can be performed to assets of type AS.MEM making them unavailable for trusted partitions. <u>Application Note:</u> For processing time on a CPU or CPU core (AS.TIME) this PP does not specify any SFRs since these depend on the technique(s) employed for time-slicing. These requirements are specified in the different modules, at least one of which is mandatory (compare Section 1.4.2). Each of these modules provides a sufficiency rationale that shows how the SFRs from that module meet the objective OT.AVAILABILITY for the asset AS.TIME. **FIA_UID.1** ensures that partitions identify themselves against the TOE before TSF-actions may be performed and, thereby, provides the basis for the access control to memory and the CPU usage time allocation described in PP modules.

Table 4: SFRs rationale.

6.3 Security Functional Requirements Dependencies Analysis

Table 5 provides the dependency analysis for the SFRs.

SFR	Dependencies	Satisfied (Y/N)
FDP_ACC.2/MA	FDP_ACF.1	Y (FDP_ACF.1/MA)
FDP_ACF.1/MA	FDP_ACC.1 FMT_MSA.3	Y (FDP_ACC.1/MA) Y (FMT_MSA.3/MA)
FMT_MSA.1/MA	[FDP_ACC.1 or FDP_IFC.1] FMT_SMR.1 FMT_SMF.1	Y (FDP_ACC.1/MA) N – The TOE does not implement roles. The entities accessing the resources are trusted partitions that do not play different roles in the access to such resources. Y (FMT_SMF.1)
FMT_MSA.3/MA	FMT_MSA.1 FMT_SMR.1	Y (FMT_MSA.1/MA) N – The TOE does not implement roles. The entities accessing the resources are trusted partitions that do not play different roles in the access to such resources.
FIA_UID.1	None	N/A
FMT_SMF.1	None	N/A

Table 5: SFRs dependency rationale.

6.4 Security Assurance Requirements

The security assurance requirements for the TOE are taken from CC Part 3 and are EAL4 augmented with ALC_FLR.2. These components are listed in Table 6.

Assurance class	Assurance component
ADV	ADV_ARC.1
	ADV_FSP.3
	ADV_TDS.2
	ADV_IMP.1
AGD	AGD_OPE.1
	AGD_PRE.1
ALC	ALC_CMC.4
	ALC_CMS.4
	ALC_DEL.1
	ALC_DVS.1
	ALC_LDC.1
	ALC_TAT.1

Assurance class	Assurance component
	ALC_FLR.2 (Augmentation)
ASE	ASE_INT.1
	ASE_CCL.1
	ASE_SPD.1
	ASE_OBJ.2
	ASE_ECD.1
	ASE_REQ.2
	ASE_TSS.1
ATE	ATE_COV.2
	ATE_DPT.1
	ATE_FUN.1
	ATE_IND.2
AVA	AVA_VAN.3

Table 6: Security Assurance Requirements.

6.5 Rationale for Security Assurance Requirements

The current PP is claimed to be conformant with the assurance package EAL4 augmented by the assurance component ALC_FLR.2.

EAL4 allows the vendor to evaluate their products at a detailed level. The chosen assurance level is appropriate for the threats defined for the environment.

The augmentation of ALC_FLR.2 was chosen to give greater assurance of the developer's continuous flaw remediation processes.

Application Note: The additional management effort to be implemented by vendors for satisfying ALC_FLR.2 is typically easily achieved and insignificant compared to the increase of assurance it provides. Furthermore, it adequately fits the level of assurance expected for the type of TOE covered by this Base PP. ALC_FLR.3 is not chosen as the increase of vendor's management effort does not bring commensurate benefits in terms of assurance compared to ALC_FLR.2.

6.6 Security Assurance Requirements Dependencies Analysis

The set of SARs included in this PP is the one associated to the EAL4 assurance package, whose internal dependencies between SARs are satisfied, plus the augmentation ALC_FLR.2 which does not depend on any other SAR. Therefore, all the dependencies for the selected set of SARs are satisfied.

Chapter 7 Summary and Conclusion

This PP provides the basis for the CC evaluation of Separation Kernel TOEs. Their evaluation may be described by STs claiming conformance to this PP and, potentially, additional PP modules.

Chapter 8 List of Abbreviations

Abbreviation	Translation
CC	Common Criteria
HW	Hardware
MILS	Multiple Independent Levels of Security
OS	Operating System
PP	Protection Profile
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SK	Separation Kernel
SSP	System Security Policy
SW	Software
TOE	Target of Evaluation
TSF	TOE Security Function

Table 7: Abbreviations.

Chapter 9 Bibliography

[CC1] Common Criteria for Information Technology Security Evaluation. Part 1: Introduction and general model. Version 3.1, Revision 5. April 2017. CCMB-2017-04-001

[CC2] Common Criteria for Information Technology Security Evaluation. Part 2: Security Functional Components. Version 3.1, Revision 5. April 2017. CCMB-2017-04-002

[CC3] Common Criteria for Information Technology Security Evaluation. Part 3: Security Assurance Components. Version 3.1, Revision 5. April 2017. CCMB-2017-04-003

[CC4] Common Criteria for Information Technology Security Evaluation. Evaluation Methodology. Version 3.1, Revision 5. April 2017. CCMB-2017-04-004

[DSC] CCDB DSC WG, "Dedicated Security Components (DSC) Essential Security Requirements Version 1.2," May 2016. [Online]. Available: http://www.commoncriteriaportal.org/communities/CCDB_DSC_ESR_v1.2.pdf.

[TM1] Period-based scheduling module. certMILS Collaboration. Version 1.0. May 2018.

[TM2] Priority-based scheduling module. certMILS Collaboration. Version 1.0. May 2018.

[TM3] CPU-affinity-based allocation module. certMILS Collaboration. Version 1.0. May 2018.