Chapter 1 Introduction

This section identifies the PP module as well as the base PP and provides a module overview for potential users.

1.1 PP Module Reference

Title: MILS Platform Protection Profile Secure Boot Module
Sponsor: certMILS Consortium
CC Version: 3.1 (Revision 5)
Assurance Level: see the Base PP.
Version: draft
Keywords: Base PP, PP Module, Operating System, Separation Kernel, MILS

1.2 Base PP Identification

Base MILS Platform Protection Profile, Version: 1.0

1.3 PP Module Overview

This module supplements the base PP by specifying the minimum functionality for secure boot process that a TOE has to provide.

Chapter 2 Consistency Rationale

This section states the correspondence between the PP module and its base PP.

2.1 TOE Type Consistency

The TOE type for which both the base PP and this PP module are designed is “a special kind of operating system, namely an SK.”

An SK is a special kind of operating system that allows to effectively separate different containers called “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).

The PP module extends the base PP by specifying security objectives covering functions relative to a secure update process.

2.2 Security Problem Definition Consistency

2.2.1 Assets

The base PP describes the following assets to be protected:

- Memory (AS.MEM)
- CPU time (AS.TIME)

This PP module does not add any asset.

2.2.2 Threats

The base PP describes the following threats contemplated:

- T.DISCLOSURE
- T.MODIFICATION
- T.DEPLETION

This PP module contemplates the following additional threats:

- T.ALTER_BOOT_IMAGE

The threat T.ALTER_BOOT_IMAGE brings a specific compatible scenario associated to T.MODIFICATION.

2.2.3 Organizational Security Policies

Neither the base PP nor this PP module define organizational security policies.

2.2.4 Assumptions

This PP module defines the following additional assumptions:

- A.SECURE_ANCHOR

This additional assumption is compatible with the assumptions defined in the base PP. The assumptions included in the base PP are applicable with no changes.

2.3 Security Objectives Consistency

The base PP describes the security objectives to be implemented:

- OT.CONFIDENTIALITY
- OT.INTEGRITY
- OT. AVAILABILITY

This PP module adds the following security objectives for the TOE:

- OT.DETECT_MODIFICATION

This security objective adds security functionality to the TOE regarding the secure boot process which is compatible to the rest of security objectives for the TOE defined in the base PP.

2.4 Security Functional Requirements Consistency

In addition to the set of SFRs included in the base PP, this PP module defines:

- FPT_TIM.1 TSF Integrity Monitoring and Action – This SFR is compatible with the set of SFRs defined in the base PP as it adds independent and specific functionality for checking the integrity of the data used for the secure boot process.
- FMT_SMF.1 Specification of Management Functions – This SFR is compatible with the set of SFRs defined in the base PP, as it extends the SFR FMT_SMF.1 of the base PP with additional management functionality.
- FMT_MTD.1 Management of TSF Data – This SFR is compatible with the set of SFRs defined in the base PP as it adds independent and specific management functionality associated to the data used for the authenticity and integrity verification of the boot process.
- FDP_DAU.1 Basic Data Authentication – This SFR is compatible with the set of SFRs defined in the base PP as it adds independent and specific functionality for authenticating the data used for the secure boot process.

Chapter 3 Conformance Claim

This protection profile module 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 [1]
- Common Criteria for Information Technology Security Evaluation. Part 2: Security Functional Components. Version 3.1, Revision 5. April 2017. CCMB-2017-04-002 [2]
- Common Criteria for Information Technology Security Evaluation. Part 3: Security Assurance Components. Version 3.1, Revision 5. April 2017. CCMB-2017-04-003 [3]

as follows

- Part 2 extended.

The “Common Methodology for Information Technology Security Evaluation, Evaluation Methodology; Version 3.1, Revision 5, April 2017. CCMB-2017-04-004, [4]” has to be taken into account.

This protection profile module is associated with the Base MILS Platform Protection Profile Version 1.0.

3.1 Conformance Rationale

Since a PP module cannot claim conformance to any protection profile, this section is not applicable.

3.2 Conformance Statement

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

Note: claiming conformance to this PP module also requires claiming conformance to the Base MILS Platform Protection Profile.

Chapter 4 Security Problem Definition

This section describes the security aspects of the environment in which the TOE claiming conformance with the PP module 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.
- Organizational security policies with which the TOE must comply.
- Assumptions about the secure usage of the TOE.

4.1 Threats

The threat countered by this PP module is the threat of loading software and data during the boot process that have been altered in an unauthorized way, thereby allowing an attacker to compromise the security of the system completely.

Threat Agent

The threat agent is any untrusted subject that has the ability to modify the boot image (i. e. the code and data) loaded as part of the boot process. To do this the boot image (or at least parts of it) must be stored in a form that allows such modifications outside of the secured development environment of the boot image. If a TOE has its boot image stored in a way that it cannot be altered outside of the secure development environment or without violating an assumption made on the TOE environment, this PP module is not applicable.

Threats agents are therefore subjects that attempt to use the TOE functions to perform the manipulation of the boot image. While it is still assumed that the TOE does not directly allow untrusted entities to perform such manipulations, the TSF may have flaws that an untrusted attacker could use to bypass those protection mechanisms.

T.ALTER_BOOTIMAGE

An attacker may alter the software and/or data loaded from modifiable persistent storage or from other sources where modifications are possible as part of the boot process in an unauthorized way, allowing the attacker to take over the system. Note: the boot process may consists of several steps where step n-1 checks the integrity of the software and data loaded for step n before loading it and passing control to it.

4.2 Organizational Security Policies

This module defines no organizational security policies.

4.3 Assumptions

A.SECURE_ANCHOR

The initial part of the boot process (step 1) is performed using a secure anchor which is not modifiable by the threat agent defined above. When the initial part of the boot process relies on data that needs to be kept confidential, the secure anchor also ensures that this information cannot be obtained by the assumed threat agent.

Chapter 5 Security Objectives

Security objectives are concise, abstract statements of the intended solution to the problem defined by the security problem definition (see previous section). The set of security objectives for a TOE form a high-level solution to the security problem. This high-level solution is divided into two part-wise solutions: the security objectives for the TOE, and the security objectives for the TOE's operational environment.

This section presents the solution to the security problem in terms of objectives for the TOE and its operational environment.

5.1 Security Objectives for the TOE

OT.DETECT_MODIFICATION

A TOE compliant with this PP module provides a step-by-step boot process where step n-1 verifies the integrity of the software and data loaded as part of step n of the boot process before passing control to it.

5.2 Security Objectives for the Operational Environment

OE.SECURE_ANCHOR

The first step of the boot process is performed by software and data that cannot be altered by the threat agent as defined in chapter 4 of this document.

OE.CONFIDENTIAL_ANCHOR

Any data used as part of the first step of the boot process that has to be kept confidential to the threat agent is kept in a storage that the threat agent cannot read even when the TSF of the TOE has been compromised.

5.3 Security Objectives Rationale

A secure boot process has to ensure that the integrity of any software and data loaded as part of this process is verified before control is passed to that software. This can be done in a step-by-step process where step n-1 verifies the integrity of the software and data loaded in step n before passing control to it. Since the integrity of the software and data loaded in the first step cannot be verified by itself, this software and data need to be part of an environment that the assumed attacker cannot modify (e. g. by having it in ROM in a system that is protected from physical access by the assumed thread agent).

Quite often the integrity verification is based on cryptographic mechanisms that require some information like keys to be kept confidential. This information then has to be stored in a location that the assumed threat agent cannot read or modify without violating the assumption of a physically protected environment.

Chapter 6 Extended Components Definition

This section includes possible extended functional components definitions where new functional components not included in CC Part 2 are introduced.

6.1 FPT_TIM TSF Integrity Monitoring

6.1.1 Family Behaviour

FPT_TIM.1 is identical to FDP_SDI.2 defined in the CC except that it applies to TSF and TSF data.

6.1.2 Component Levelling

The FPT_TIM family contains only one component: FPT_TIM.1.

FPT_TIM.1 is, therefore, not hierarchical to any other component within the FPT_TIM family.

6.1.3 Management

See management description specified for FDP_SDI.2 in [CC].

6.1.4 Audit

See audit requirement specified for FDP_SDI.2 in [CC].

6.1.5 FPT_TIM.1 TSF Integrity Monitoring and Action

Hierarchical to: No other component

Dependencies: No dependencies

FPT_TIM.1.1 The TSF shall monitor [selection: TSF code, TSF data, [assignment: parts of TSF code, parts of TSF data]] for [assignment: integrity errors] using the following rules: [assignment: rules that define how the integrity is verified].

FPT_TIM.1.2 Upon detection of a data integrity error, the TSF shall [assignment: action to be taken].

6.1.6 Rationale

[CC] defines integrity verification for user data. This SFR extends this functional claim to TSF data and TSF code.

Chapter 7 Security Requirements

This section defines the Security Functional requirements (SFRs). This PP module does not introduce specific assurance requirements. The assurance requirements are defined by the Base MILS Platform Protection Profile.

7.1 Security Functional Requirements

7.1.1 Mandatory SFR: Protection of the TSF

7.1.1.1 FPT_TIM.1 TSF Integrity Monitoring and Action

FPT_TIM.1.1 The TSF hosted in a non-modifiable environment or whose integrity has already been verified shall monitor [*the next part of the TSF code and TSF data to be loaded and executed as part of the TSF*] for [unauthorized modification of that TSF code and TSF data] using the following rules:

[selection:

- cryptographic keyed message digest compliant with [assignment: keyed message algorithm according to a defined standard] calculated over the TSF and TSF data to be loaded;
- digital signature using [assignment: digital signature algorithm according to a defined standard] of the TSF and TSF data;
- [assignment: other Integrity verification mechanism]
-].

FPT_TIM.1.2 Upon detection of a data integrity error, the TSF shall [stop the boot process].

Application Note: if a cryptographic function is used for integrity verification this function also has to be defined as a SFR within the Security Target including the SFRs that define how cryptographic keys are generated or imported and the SFRs defining the protection of those keys.

Application Note: When the TSF detects an unauthorized modification the TSF as a whole shall stop execution. It is up to the individual TOE to define a process how to overcome this situation. In some cases the TSF may implement a 'shortened' boot process which results in some maintenance mode where a trusted role is able to 'repair' the TSF. In other cases the repair operation may be performed totally by functions within the TOE environment.

7.1.2 Optional Management SFRs in Case the Integrity Database can be updated

7.1.2.1 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions: [updating TSF code and data that is part of the secure boot process while maintaining the ability to verify the integrity of that updated TSF code and data as part of the secure boot process].

7.1.2.2 FMT_MTD.1 Management of TSF Data

FMT_MTD.1.1 The TSF shall restrict the ability to [*generate and update*] the [integrity data base of the TSF and TSF data for the integrity verification of the TSF and TSF data] to [a process operating in maintenance mode of the TOE as part of the secure boot process].

Application Note: There are multiple possible ways how the integrity database can be updated by the TSF. This can either be an explicit action of an authorized role (which can be an authorized trusted partition) or it can be an action requested by an external entity over the network.

In any case the following conditions must be met:

- The update process needs to be performed as part of the secure boot process, i. e. it must be performed before any part of the TSF and data is loaded that is not part of the secure boot process
- The update process needs to be performed as part of a maintenance process ensuring that the update is consistent with the rest of the TSF.

7.1.2.3 FDP_DAU.1 Basic Data Authentication

FDP_DAU.1.1 The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of [TSF code and TSF data to be updated].

FDP_DAU.1.2 The TSF shall provide [the TSF already loaded and started as part of the secure boot process] with the ability to verify evidence of the validity of the indicated information.

Application Note: the criteria for the validity need to include evidence that the update of the TSF code and TSF data comes from an authorized source. The data used later for the verification of the integrity of this updated part of the TSF code and data may either be part of that data (e. g. as a digital signature) or may be generated by the update process. If this is the case the process to generate and protect the data used for integrity verification within the secure boot process needs to be defined in a Security Target claiming compliance to this PP module with its optional SFRs.

7.2 Security Requirements Rationale

The purpose of this PP module is the provision of a secure boot process that ensures that code and data loaded during the boot process has not been tampered with.

The functions provided by this PP module may well be used for higher level functionality like the implementation of cryptographic protocols. If a TOE does this, it needs to specify the SFRs for those protocols in addition to the SFRs defined in this PP module.

FPT_TIM.1 addressed the security objective OT.DETECT_MODIFICATION by requiring that – starting from a non-modifiable anchor – the boot process is able to detect modifications for the code and data loaded as the next step in the secure boot process. By induction this shows that the integrity of the whole code and data covered by the secure boot process is verified.

The optional SFRs ensure that the objective OT.DETECT_MODIFICATION is also met when a part of the TSF code and data covered by the secure boot process is updated. SFRs FMT_SMF.1 and FMT_MTD.1 ensure that such an update can only be performed when requested by an authorized entity and only as part of the secure boot process thereby prohibiting any interference of the update process by untrusted software. The SFR FDP_DAU.1 ensures that the integrity of the updated TSF code and data can be verified as part of the next secure boot process performed for the TOE.

7.3 Security Functional Requirements Dependencies Analysis

The following dependencies are defined for the SFRs used in this PP module:

SFR	Dependencies	Satisfied?
FPT_TIM.1	none	yes
FMT_SMF.1	none	yes
FMT_MTD.1	FMT_SMR.1 FMT_SMF.1	no yes

SFR	Dependencies	Satisfied?
FDP_DAU.1	none	yes

The dependency of FMT_MTD.1 on FMT_SMR.1 is not satisfied since the update may be triggered in a way that does not involve a role of the TOE. For example the update may be triggered by an external signal or an automatic check where the TOE identifies that an update to TSF code and data is available and then performs the update as part of the secure boot process. In this case no TSF-defined role is involved in this process.

Chapter 8 Application Notes

A secure boot process ensures that the software and data loaded as part of the boot process is genuine as defined and delivered by the TOE developer and has not been tampered with before it is loaded and control is passed to it. This needs to be guaranteed even in the case a threat agent has overcome the TOE protection features for the protection of persistent storage that is used to store the code and data loaded as part of the boot process.

Such a process requires therefore a ‘secure anchor’ that is non-modifiable within the assumed TOE environment. This ‘secure anchor’ will perform the first step of the secure boot process and verify the code and data loaded for the next step. Such a secure anchor can be implemented by code and data stored in ROM or other forms of persistent storage that require physical access to be modified.

This PP module does not prescribe how the integrity verification is performed. It just requires that this process cannot be forged by a threat agent which requires that it is not possible for the threat agent to deliberately create a modified version of a part of the code and data loaded during the boot process that passes the integrity verification mechanism without such modifications being detected.

In many cases the integrity verification mechanism will rely on some secret like a cryptographic key to satisfy the condition mentioned above. In this case this secret also needs to be kept confidential such that a threat agent is not able to deduce information about the value of that secret.

Chapter 9 List of Abbreviations

Abbreviation	Translation
CC	Common Criteria
PP	Protection Profile
HW	Hardware
SW	Software
OS	Operating System
SFR	Security Functional Requirement
SAR	Security Assurance Requirement

Chapter 10 Bibliography

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

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

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

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