The standard for
secure digital services
and devices

GlobalPlatform Technology

SESIP Profile for Secure MCUs and MPUs

Version 1.0.0.4 [target v1.1]

Public Review

September 2024

Document Reference: GPT_SPE_150

Contents

Tables

Figures

1 Introduction

The Security Evaluation Standard for IoT Platforms ([SESIP]) specifies general requirements for Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs) designed to be used in the evaluation and certification of IoT products. A SESIP Profile is a security profile generic to a type of platform (part), defining the SESIP requirements in terms of security features and evaluation activities to be addressed during the evaluation of a platform (part) of the type targeted by the profile.

This document provides a SESIP Profile for security evaluations of Microcontroller Units (MCUs) and Memory Protection Units (MPUs).

1.1 Audience

This document is intended primarily for the use of all stakeholders involved in MCU/MPU security evaluations, including developers basing their own product on evaluated MCUs/MPUs.

Laboratory customers, regulatory authorities, organizations and schemes using peer assessment, accreditation bodies, and others use this document in confirming or recognizing the security level of MCU/MPU platform parts.

The use of this document will allow harmonization and comparability between MCU/MPU security evaluations. The acceptance of results between schemes is facilitated if laboratories conform to this document.

1.2 IPR Disclaimer

Attention is drawn to the possibility that some of the elements of this GlobalPlatform specification or other work product may be the subject of intellectual property rights (IPR) held by GlobalPlatform members or others. For additional information regarding any such IPR that have been brought to the attention of GlobalPlatform, please visit https://globalplatform.org/specifications/ip-disclaimers/. GlobalPlatform shall not be held responsible for identifying any or all such IPR, and takes no position concerning the possible existence or the evidence, validity, or scope of any such IPR.

1.3 References

This section lists references applicable to this specification. The latest version of each reference applies unless a publication date or version is explicitly stated.

Table 1-1: Normative References

Standard / Specification	Description	Ref
GP_FST_070	GlobalPlatform Technology Security Evaluation Standard for IoT Platforms (SESIP) Methodology v1.2, July 2023	[SESIP]

1.4 Terminology and Definitions

Selected terms used in this document are included in Table 1-2. Additional terms are defined in [SESIP].

Table 1-2: Terminology and Definitions

Term	Definition
IoT Platform	In the context of this document, identical to Connected Platform, as defined in [SESIP].
Secure Enclave	A separate self-contained execution environment that is in exclusive control of its resources (e.g. Secure Element, SoC Secure Sub-System, TPM, or TEE) and that implements dedicated hardware protections.
SESIP Profile (SP)	A security profile generic to a type of platform (part). Equivalent to CC Protection Profile: A generic SESIP Security Target defining the SESIP requirements in terms of security features and evaluation activities to be addressed during the evaluation of a platform (part) of the type targeted by the profile.

1.5 Abbreviations

Table 1-3: Abbreviations

Abbreviation	Meaning
MCU	Microcontroller Unit
MPU	Micro Processor Unit
RoT	Root of Trust
SAR	Security Assurance Requirement
SESIP	Security Evaluation Standard for IoT Platforms
SFR	Security Functional Requirement
SP	SESIP Profile
ST	Security Target
TEE	Trusted Execution Environment
TOE	Target Of Evaluation
TPM	Trusted Platform Module

1.6 Revision History

GlobalPlatform technical documents numbered *n*.0 are major releases. Those numbered *n*.1, *n*.2, etc., are minor releases where changes typically introduce supplementary items that do not impact backward compatibility or interoperability of the specifications. Those numbered *n*.*n*.1, *n*.*n*.2, etc., are maintenance releases that incorporate errata and clarifications; all non-trivial changes are indicated, often with revision marks.

Table 1-4: Revision History

Date	Version	Description
Oct 2021	1.0	Public Release
Apr 2024	1.0.0.2	Member Review
Sep 2024	1.0.0.4	Public Review
TBD	v1.1	Public Release Updated to v1.2 of GlobalPlatform SESIP specifications. Changes introduced in this version are editorial and do not impact backward compatibility.

2 SESIP Profile Introduction

This SESIP Profile (SP) describes a generic MCU/MPU component that may be used in various IoT use cases. It describes an essential set of basic security properties that are common to IoT use cases and that shall be consistently evaluated (see section 4.2, "Base SP Security Functional Requirements"), and defines further security feature packages to cover most IoT platform use cases (see section 2.2, "Platform Component Functional Overview and Description").

MCUs/MPUs are intended to be integrated into a Connected Platform (as defined in [SESIP]), also called IoT Platform in this document.

The MCU/MPU is the lowest level building block of an IoT Platform and therefore is intended to provide the fundamental security service layers of the platform; this includes the immutable Root of Trust (RoT). In particular, these security services enable the higher layers to trust, manage, and update the state, software, and configuration of the MCU/MPU; control access to the device on the lowest layer; store assets and perform secure and cryptographic operations with them; attest the secure state of the device at start-up and during runtime; and provide different levels of isolation and protection such that it is possible to shield different types of operations and computation.

MCUs/MPUs may also require extended security features (e.g. strong cryptography or secure communications) and/or could be operating in different environments (e.g. publicly accessible or in a private environment; open or closed to untrusted software downloads); these different use cases involve specific security features. To allow the evaluation of these use cases, this SESIP Profile is made of a Base SESIP Profile plus Packages that can be added depending on the functionality and/or the product environment context.

Some MCUs/MPUs may only target restricted use cases where additional measures (e.g. physical access policies or trustworthiness of the code installed on the MCU/MPU) are enforced by the operational environment under the responsibility of the appropriate entities.[1] These restrictions are possible and are handled in section 2.2.4, "Use Case".

SESIP security requirements cover security objectives for the final platform; typically, an MCU/MPU supports the implementation of such security requirements up to a certain level and sometimes completely. When a security requirement is partially implemented, the MCU/MPU composition guidance describes how to securely complete the objective on the upper layer(s).

Other security concerns related to MCU/MPU development and maintenance in the field (e.g. secure life cycle, secure provisioning, or secure testing) are also considered in this SESIP Profile.

Moreover, SESIP methodology requires that the entire MCU/MPU functionality does not compromise the security claims.

If an MCU/MPU is certified compliant with this SESIP Profile, customers can rely upon these security features and upon the security assurance to build a complete IoT Platform if they follow the user guidance documentation provided with the certified MCU/MPU. In particular, it is important to carefully verify the assumptions about the use case (see section 2.2.4, "Use Case") and to read the objectives for the environment (in section 3, "Security Objectives for the Operational Environment") and evaluate whether these are in line with the consumers' operational environment.

[1] Entity that uses or interacts with the IoT device (e.g. owner, end user, OEM, service provider, integrator, or administrator) and is responsible for the usage of the product in the expected operational context.

This SESIP Profile contains guidance paragraphs:

- AN: Guidance that shall be considered and followed for Security Target (ST) writing
- INFO: Additional context information

2.1 SESIP Profile Reference

SP Name	SESIP Profile for Secure MCUs and MPUs
SP Version	v1.1
TOE Type	Secure Processing Unit for IoT devices

2.2 Platform Component Functional Overview and Description

AN The following description shall be adapted/replaced in a Security Target by a description of the actual TOE.

The TOE consists of hardware and software. The software is usually divided into an immutable part, stored in ROM, which typically includes at least the code performing the MCU/MPU's secure initialization; and the firmware, which can be modified and updated during the product life cycle.

The TOE is intended to be used by an integrator as a basis to develop an IoT Platform by adding to it the required components, including a Root of Trust software layer, an operating system, and connectivity, as well as additional hardware components as required by the use case.

2.2.1 Platform Security Features and Scope

The typical core security features of the TOE are:

- Secure initialization, to control the platform's initialization sequence
- Secure update and residual information purging for life cycle handling
- Secure debugging in case of investigation need; optionally auditing and logging features can also be implemented

Optional packages may be selected depending on the context of use of the MCU/MPU:

- Cryptographic operations, often based on hardware cryptographic accelerators
- Hardware protections to handle hostile environments
- Isolation mechanisms controlling access between different parts of the software building the product
- Secure storage functionality provided to the application

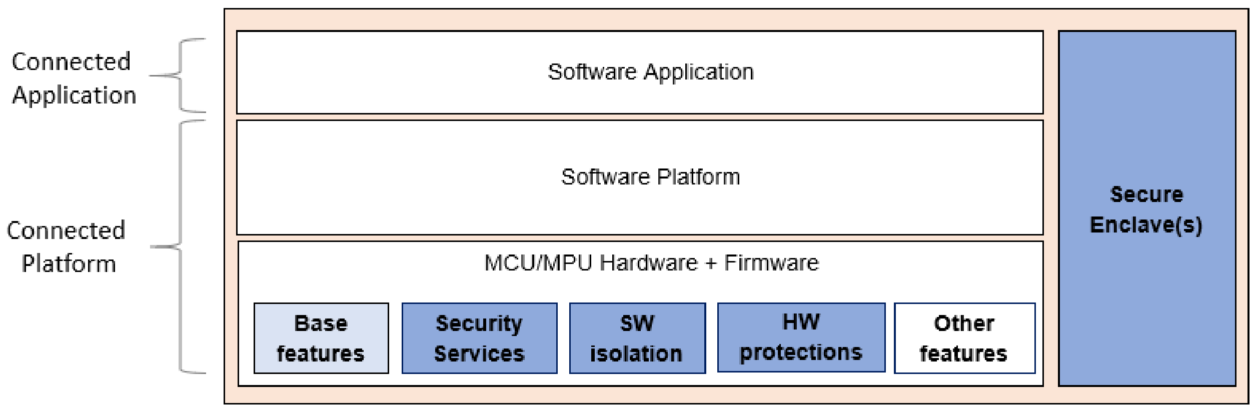
This core profile can also be extended with other optional features such as:

- Secure handling of applications
- Services such as secure communications and attestation

The physical scope includes only the processing unit itself and the logical scope includes only the MCU/MPU's firmware. Additional hardware parts included in the platform together with their firmware, as well as higher software layers (e.g. software Root of Trust services, software platform parts, or final applications) are outside of the evaluation scope.

Figure 2-1 illustrates an example of the MCU/MPU scope and integration in a final product. The **light blue parts are within the base evaluation scope**, the **darkest blue parts are optional**, and the white parts are outside of the evaluated scope.

Figure 2-1: TOE Scope Representation Example



AN The Security Target shall precisely describe the physical and logical scopes of the TOE, and provide all necessary details required to identify the libraries, drivers, and other modules that are used, together with their versions and any other relevant information. Additional details about the security features supported by the MCU/MPU must be described.

AN The tables below provide typical examples of scope description content that must be adapted in the Security Target to reflect the targeted MCU/MPU:

The TOE provides the following hardware components and interfaces:[2]

Table 2-1: Hardware Components and Interfaces of the TOE

Component/Interface	Description	Identification
Integrated Circuit		(hardware ID(s))
Communications ports	*e.g. ISO7816/i2c/UART/ISO1443/SWP/SPI/ USB/GPIOs/Ethernet/Secure Mailbox*	
RF interfaces	*e.g. Wi-Fi, Bluetooth, NFC*	
Debug ports	*e.g. JTAG*	
Memories	*e.g. internal RAM, ROM, Flash, OTP*	
Secure Enclave	System characteristics	
…		

[2] Hardware interfaces are hardware modules and physical ports exposed to the environment.

The TOE provides the following software components and interfaces:[3]

Table 2-2: Software Components and Interfaces of the TOE

Component/Interface	Description	Identification / Version
Bootloader	ROM image	
Firmware	FLASH image	
CryptoLib	FLASH image	
APIs	APIs	
…		

2.2.2 Life Cycle

AN The Security Target shall describe the life cycle of the MCU/MPU under evaluation.

The description must present an overview of the main phases from the hardware and software design to the product end-of-life; for each phase, all possible MCU/MPU state(s) must be identified. How transitions between those states are secured must also be explained.

The description must identify all Roots of Trust integrated into the MCU/MPU (e.g. for secure boot or for data storage); for each RoT, it shall be specified in which phase and under which MCU/MPU state the integration is performed, and how the integration is secured.

The description must include how the provisioning is performed (directly in the ST or referencing the appropriate guidance document).

[3] Software interfaces are the logical features exposed to the environment (including firmware).

2.2.3 Configurations

AN This SESIP Profile is intended to cover a wide variety of MCU/MPUs covering different use cases and feature sets. The Security Target must indicate the implemented configuration in order to define the Security Functional Requirements to be claimed in subsequent sections. The following descriptions are to be completed/replaced by the specific descriptions of the TOE.

INFO A base TOE implements the core minimum security features for MCU/MPUs, so must meet the security requirements defined in section 4.2, “Base SP Security Functional Requirements”.

The MCU/MPU ensures the execution of platform trusted code, particularly the functions related to secure boot, updatability, and code isolation.

INFO In many cases, cryptographic extensions will be implemented, in which case the TOE must meet the additional security requirements defined in section 4.3, “Package ‘Security Services’ Security Functional Requirements”.

[Optional] The security features discussed above are complemented by security services intended to be used by the higher software layers to implement a full-fledged Root of Trust and operating system.

INFO In some cases, the security requirements can be fulfilled in a Secure Enclave. A Secure Enclave is a separate self-contained execution environment that is in exclusive control of its resources (e.g. Secure Element, SoC Secure Sub-System, TPM, or TEE) and that implements dedicated hardware protections.

INFO Typically, secure storage functionality will be implemented, in which case the TOE must meet the additional security requirements defined in section 4.7, “Package ‘Secure Storage’ Security Functional Requirements”.

AN The Security Target must indicate whether some of the security requirements are fulfilled by using a Secure Enclave. If so, these security requirements must be moved to section 4.6, “Package ‘Secure Enclave’ Security Functional Requirements”. If a security feature is present both in the Secure Enclave and outside the Secure Enclave, the claims are to be duplicated.

[Optional] A Secure Enclave is used to fulfil the following security features: <security features>.

2.2.4 Use Case

INFO The base target of this SESIP Profile is executing in a non-hostile environment **[trusted user only]** and does not allow the download of untrusted software **[trusted code only]**. However, for most use cases either **[any user]**, **[any code]**, or **both** must be assumed, to cover execution in hostile environments and/or execution of untrusted software.

Note that in the context of this profile, trustworthiness of the code is only related to the confidence or not that the code could harm the security features claimed by the platform; “untrusted” code may be from any party including the first party, for instance when the code has not been subjected to a security analysis or has been evaluated at a lower level than the MCU/MPU.

Such environmental conditions have an impact on the security requirements (see section 4.2, “Base SP Security Functional Requirements”) to be implemented by the TOE.

AN The Security Target must clearly identify the environmental conditions under which the TOE, due to its specific implementation, can be securely used. The Security Target must specify:
1) whether **[trusted user only]** or **[any user]** can physically access the product and
2) whether **[trusted code only]** or **[any code]** can be run on the product.

INFO The **[* only]** configurations described below are use case restricted, where the security needs depend on the operational environment and/or TOE features; it is the user’s responsibility to ensure that the operational environment and/or TOE features are consistent with the assumptions.

The **[any *]** configurations described below implement self-contained security measures and can be used in any environment.

[trusted user only]

The product may be operated in controlled environments to protect against attackers other than the device’s legitimate owner. A controlled environment is typically enforced by strong organizational policies and means, for instance in the case of industrial use, or by the proper protection of the private environment of the end-user.

[any user]

Or the product may be physically accessed by an unknown or untrusted user, in an environment where access to the product cannot be sufficiently controlled or even in a more hostile environment.

INFO If the use case analysis is not fully conclusive or risk assessment shows non-negligible residual risk, then it is strongly recommended to assume operation in a potentially uncontrolled environment (**[any user]**).

[trusted code only]

The use case environment and the product setup may ensure that the final IoT product executes only trusted code from the product developer and that other entities cannot load untrusted code onto the product.

[any code]

Or it cannot be excluded that the product will execute code that is unknown to the product developer.

INFO If the use case analysis is not fully conclusive or risk assessment shows non-negligible residual risk, then it is strongly recommended to assume the possibility that unknown code (**[any code]**) may be executed on the microcontroller.

3 Security Objectives for the Operational Environment

For the platform to fulfil its security requirements, the operational environment (technical or procedural) must fulfil the following objectives.

- The operating system or application code is expected to verify the correct version of all platform components that it depends on, as described in the <*Reference to documentation where this is described*>.
- The operating system or application code is expected to make use of the <*Name of the secure boot*> feature as described in the <*Reference to documentation where this is described*>.

AN If the Security Target declares Package 'Hardware Protections', then the following objective for the operational environment must be included in the Security Target.

- The integrating environment is expected to configure the debug functionality as described in <*Reference to documentation where this is described*> to meet the extra physical attacker resistance.

AN If the Security Target does not declare Package 'Hardware Protections', then the following objective for the operational environment must be included in the Security Target, when needed. Protections against physical attacks may not be relevant depending on the use case; e.g. if the lifetime of the IoT product is shorter than any physical attacks would require, if there is no asset reachable by or no security impact associated to physical attacks, etc. Reasons for not needing operational environment protections against physical attacks must be described in the "Use Case" section.

- The operational environment must protect the TOE against physical access of attackers as described in <*Datasheet] section "limitations"*>.

AN If the Security Target does not declare Package 'Software Isolation', then the following objective for the operational environment must be included in the Security Target.

- The operational environment must not allow the deployment of untrusted code as described in <*[Datasheet] section "limitations"*>. That means that all code running on the product is known to the product vendor and the product vendor can confirm that the code cannot harm the claimed security functionalities.

4 Security Requirements and Implementation

AN In this chapter, the Security Target must define the Security Assurance Level claimed and the list of Security Functional Requirements in the scope of the evaluation; this list must address all security features listed in the TOE description.

In this chapter the **platform** is the **TOE** as it has been defined in section 2.2.

4.1 Security Assurance Requirements

The claimed assurance requirements package is **SESIP2** as defined in [SESIP].

This level can be raised depending on Packages selected; see Annex A for details.

4.1.1 Flaw Reporting Procedure (ALC_FLR.2)

In accordance with the requirement for a flaw reporting procedure (ALC_FLR.2), including a process to report flaws and generate and distribute any needed updates, the developer has defined the following procedure:

<Describe the procedure, including where flaws and security incidents can be reported (website and/or email address), how the reported flaws are handled in a timely manner, and how an application developer/end-user can get informed of the update. If the “Secure update of platform” SFR is removed, you have to provide a strong argumentation here why the platform is not worth getting an update. However, the process to receive the reports of flaws and handling them in a timely manner needs to be described in any case.>

4.2 Base SP Security Functional Requirements

As a base, the platform fulfils the following Security Functional Requirements:

INFO This section lists the minimum Security Functional Requirements (SFRs) to be implemented by the TOE. It is appropriate for use cases **[trusted user only]** and **[trusted code only]** (see section 2.2.4, “Use Case”).

INFO When mentioned in the following SFR sections, the SFR can be strikethrough if and only if the feature is not available to the users. If the feature is implemented but not accessible, the evaluation will assess that this cannot be used against any of the claimed SFRs.

4.2.1 Verification of Platform Identity

Requirement

The platform provides a unique identification of the platform, including all its parts and their versions.

AN If the final platform implements a secure backup and recovery process that includes the identification, then the identification can be part of the legitimate clone of the platform without invalidating this SFR. This must be described in the Security Target if the MCU/MPU supports cloning, and appropriate SFRs must be added.

4.2.2 Secure Initialization of Platform

Requirement

The platform ensures its integrity and authenticity during platform initialization. If platform integrity or authenticity cannot be ensured, the platform will go to <*list of controlled states*>.

AN The MCU/MPU is in charge of the secure initialization of its own firmware but, where applicable, it shall also include features that can be used by the rest of the IoT Platform to perform its own secure initialization.

4.2.3 Secure Update of Platform

Requirement

The platform can be updated to a newer version in the field such that the <*confidentiality,*> integrity and authenticity of the platform is maintained.

INFO This SFR may be removed (via strikethrough to increase the reader's awareness) only when strong argumentation is provided in ALC_FLR.2 to explain why updates are not necessary or possible for this platform.

INFO It is expected that for a specific platform, a subset of the security features will not be updateable, particularly those that are directly related to the secure initialization of the platform. Such security features are part of the immutable Root of Trust whose integrity cannot be verified, and the immutability of their code is a security measure.

AN As explained in [SESIP], what is "newer" and "older" must be defined (and be part of ALC_FLR.2) in the Security Target.

AN The [Datasheet] shall describe the secure rollback policies that are enforced by the MCU/MPU.

4.2.4 Residual Information Purging

Requirement

The platform ensures that <*list of data*>, with the exception of <*list of exceptions of data that is not erased automatically*>, is erased using the method specified in <*specification*> before the memory is used by the platform or application again and before an attacker can access it.

INFO This SFR is required for sensitive data erasing; this feature can be used directly for the MCU/MPU platform part or by platform parts on top for Decommission, Factory Reset, or Field Return features.

4.2.5 Secure Debugging

INFO This SFR may be removed (via strikethrough to increase the reader's awareness) if the feature is not available to the users; if implemented but not accessible, the evaluation will assess that this cannot be used against any of the claimed SFRs.

Requirement

The platform only provides <*list of endpoints*> authenticated as specified in <*specification*> with debug functionality.

The platform ensures that all user data stored, with the exception of <*list of exceptions*>, is made unavailable.

AN The MCU/MPU must at least provide access control to the debug functionality.

In practice, access to data by debug features is typically organized in "levels" under which all data are identically accessible. For instance, access to the "application level" will grant access to all application data. In such a case, <*exceptions*> must be set to "none" and the different levels must be identified and explained.

Another possibility is that debugging is only accessible in a "Field Return" mode for investigation purposes, which requires the secure erasing of all application data. In such a case, <*exceptions*> must also be set to "none", and the relation with the "Field Return" mode must be described; the "Field Return of Platform" SFR should also be claimed.

4.3 Package 'Security Services' Security Functional Requirements

INFO This package defines Security Functional Requirements needed for a platform that is intended to support implementation of extended security services.

INFO In this package, any of the SFRs could be strikethrough if not available to the users; if implemented but not accessible, the evaluation will assess that this cannot be used against any of the claimed SFRs.

4.3.1 Cryptographic Operation

Requirement

The platform provides <*list of cryptographic operations*> functionality with <*list of algorithms*> as specified in <*specification*> for key lengths <*list of key lengths*> and modes <*list of modes*>.

Operations	Algorithm	Specification	Key lengths	Modes
<TBD by the ST>	<TBD by the ST>	<TBD by the ST>	<TBD by the ST>	<TBD by the ST>
<TBD by the ST>	<TBD by the ST>	<TBD by the ST>	<TBD by the ST>	<TBD by the ST>

4.3.2 Cryptographic Key Generation

Requirement

The platform provides a way to generate cryptographic keys for use in <*list of cryptographic algorithms*> as specified in <*specification*> for key lengths <*list of key lengths*>.

4.3.3 Cryptographic KeyStore

Requirement

The platform provides a way to store <*list of assets, such as cryptographic keys and passwords*> such that not even the application can compromise the <*selection: confidentiality, integrity, authenticity*> of this data. This data can be used for the cryptographic operations <*list of operations*>.

AN The KeyStore shall be fully supported by the MCU/MPU firmware, and shall be made available to the rest of the IoT Platform to protect platform-level keys. If the features available to the higher platform layers and to the applications are different, then the SFR shall be duplicated and adapted.

4.3.4 Cryptographic Random Number Generation

Requirement

The platform provides a way based on <*list of entropy sources*> to generate random numbers as specified in <*specification*>.

AN This SFR shall be iterated to cover all random number generators in the scope of the secure MCU/MPU.

4.4 Package 'Software Isolation' Security Functional Requirements

INFO This package defines Security Functional Requirements needed to cover the use case where the final IoT product allows the execution of untrusted code and/or requires access restrictions to the platform features by the upper level (e.g. based on the access need). As explained in section 2.2.4, "Use Case", that code can be considered "untrusted" even when developed by the first party, for instance when this code has not been subjected to the same level of security analysis. These Security Functional Requirements must be claimed in case of configuration **[any code]** (see section 2.2.4, "Use Case").

4.4.1 Software Attacker Resistance: Isolation of Platform

Requirement

The platform provides isolation between the application and itself, such that an attacker able to run code as an application on the platform cannot compromise any other claimed Security Functional Requirements.

AN The MCU/MPU provides the basic mechanism used to isolate the IoT Platform from its application(s), typically based on the existence of a privileged execution state (user vs. system). The guidance shall include recommendations for the proper usage of this basic mechanism in the higher layers of the platform.

4.5 Package 'Hardware Protections' Security Functional Requirements

INFO This package defines the security requirements needed to cover the use case where the platform is physically accessible and thus requires protection against local attacks. These Security Functional Requirements must be claimed in case of configuration **[any user]** (see section 2.2.4, "Use Case").

4.5.1 Physical Attacker Resistance

Requirement

The platform detects or prevents attacks by an attacker with physical access before the attacker compromises any of the other functional requirements.

4.6 Package 'Secure Enclave' Security Functional Requirements

AN If the MCU/MPUs include a Secure Enclave as defined in section 2.2.3, "Configurations", then the Secure Enclave must expose a combined hardware/software interface to the rest of the MCU/MPU. Related additional security measures (such as cryptographic accelerations or one-time programmable (OTP) memory) must be claimed in this section. The Secure Enclave may be implemented using a separate CPU and may have dedicated resources; this must be described in section 2.2, "Platform Component Functional Overview and Description".

AN In addition to the security services mentioned above, at minimum the Limited Physical Attacker Resistance SFR must be claimed, showing that the Secure Enclave is providing hardware protections covering the security services claimed in the enclave.

4.6.1 Limited Physical Attacker Resistance

Requirement

The platform detects or prevents attacks by an attacker with physical access before the attacker compromises *the requirements claimed as part of the Secure Enclave*.

4.7 Package 'Secure Storage' Security Functional Requirements

INFO This package defines the security requirements needed to cover the use case where the platform provides secure storage functionality to the application.

Note that [SESIP] provides several SFRs to cover the secure storage functionality. This package aims to provide flexibility to let developers claim the suitable solution. These solutions are:

- Secure Trusted Storage: Requires integrity, authenticity, and binding to platform instance. Confidentiality is not required.
- Secure Confidential Storage: Requires confidentiality, integrity, authenticity, and binding to the platform instance. Confidentiality may be achieved by non-encryption means.
- Secure Encrypted Storage: Requires confidentiality using encryption. Also covers integrity.
- Secure Data Serialization: Covers the use case where data is stored outside the platform. The developer may choose the protection level: confidentiality, integrity, authenticity, binding to the platform instance, versioning.

AN The developer must choose at least one of the SFRs defined in this section.

INFO This SP can be used in composition with a certified TOE according to the "SESIP Profile for Secure External Memories". In this case, the Security Target must explicitly state which SFRs are covered by the certified external memory.

4.7.1 Secure Trusted Storage

Requirement

The platform ensures that all user data stored, except for <*list of data stored in plaintext*>, is protected to ensure its integrity, authenticity, and binding to the platform instance.

4.7.2 Secure Confidential Storage

Requirement

The platform ensures that all data stored by the application, except for <*list of data stored*>, is protected to ensure its confidentiality, integrity, authenticity, and binding to the platform instance.

4.7.3 Secure Encrypted Storage

Requirement

The platform ensures that all user data stored, except for <*list of data stored in plaintext*>, is encrypted as specified in <*specification*> with a platform instance unique key of key length <*key length*>.

4.7.4 Secure Data Serialization

Requirement

The platform ensures that all data stored outside the direct control of the platform, except for <*list of data stored outside the direct control of the platform*>, is protected such that the <*selection: confidentiality, integrity, authenticity, binding to the platform instance, versioning*> is ensured.

4.8 Additional Security Functional Requirements (Optional)

INFO	Other SFRs defined in [SESIP] can be part of the Security Target if implemented at the MCU/MPU level as support for IoT Platform upper layers. The inclusion of such SFRs must not impact the security of SFRs defined in the Base SP and Packages.
AN	If one of the SFRs is partially implemented at the MCU/MPU level, this must be part of composition guidance.
INFO	Additional proprietary SFRs can also be added, only if this does not affect any of the SESIP SFRs; moreover, those proprietary SFRs must be assessed by the Evaluation Laboratory and accepted by the Certification Body.

5 Mapping and Sufficiency Rationales

The claims of the Base SP apply when the secure MCU/MPU is operating in a non-hostile environment and executing only trusted software (i.e. use cases **[trusted user only]** and **[trusted code only]**).

Packages may need to be added to cover specific features and/or environments, as follows:

- If the product is intended to support extended services, Package 'Security Services' must be added to the claim.
- If the product operates in a hostile environment, use case **[any user]**, Package 'Hardware Protections' must be added to the claim.
- If the product allows the execution of untrusted software, use case **[any code]**, Package 'Software Isolation' must be added to the claim.
- If the product uses a **[Secure Enclave]**, Package 'Secure Enclave' must be added to the claim.

The table below indicates which SESIP levels are possible to claim for each of the packages:

Table 5-1: SESIP Levels for Packages

Features Claim	Minimum to maximum allowed SESIP Levels
Base SP (mandatory) *[trusted user only], [trusted code only]* Base SFRs: ○ Verification of Platform Identity ○ Secure Initialization of Platform ○ Secure Update of Platform ○ Residual Information Purging ○ Secure Debugging	SESIP2
Base SP + Package 'Security Services' (optional) *[trusted user only], [trusted code only]* Additional SFRs: ○ Cryptographic Operation ○ Cryptographic Key Generation ○ Cryptographic KeyStore ○ Cryptographic Random Number Generation	SESIP2
Base SP + Package 'Software Isolation' [4] **(optional)** *[any code] (replaces [trusted code only])* Additional SFRs: ○ Software Attacker Resistance: Isolation of Platform	SESIP2 or SESIP3

[4] To identify an attack, an attacker may use physical attack techniques on the TOE or similar devices. Therefore, considerations about attack paths regarding software and physical attack resistance shall consider this attacker capability on all assurance levels that involve an independent vulnerability analysis (SESIP2 – SESIP5). This is also included as refinement for the corresponding AVA_VAN activities.

Features Claim	**Minimum to maximum allowed SESIP Levels**
Base SP + Package 'Hardware Protections' [5] **(optional)** *[any user] (replaces [trusted user only])* Additional SFRs: o Physical Attacker Resistance	SESIP3 to SESIP5 [6]
Package 'Secure Enclave' (optional) *Use of Secure Enclave* *Applicable SFRs product dependent (to be defined in the Security Target)*	SESIP3 to SESIP5 [7]

Combination of Base SP with several Packages is allowed; in such a case, the minimum SESIP level allowed is the highest minimum level of the selected packages, and the maximum SESIP level allowed is the highest maximum level of the selected packages.

For a combination with the Secure Enclave package, this package has its own security level, which can be different from the rest of the MCU/MPU.

The independent optional security features listed in section 4.8, "Additional Security Functional Requirements (Optional)" can be added to any combination of the previous claims without affecting the SESIP level.

5.1 SESIP Level Sufficiency

As the SESIP level is directly used per [SESIP], the sufficiency is also per [SESIP].

[5] The complexity of adding protection against physical attack resistance is typically significant. Similarly, vulnerability analysis and testing tasks are typically more complex. To make this transparent and explicit to facilitate the user's risk assessment, it is expected that the activities for AVA_VAN are increased appropriately. The developer may support these activities by submitting source code to the evaluation laboratory, thereby decreasing the AVA_VAN efforts.

[6] SESIP assurance level will depend on the hardware protections in place.

[7] SESIP assurance level will depend on the Secure Enclave technology, e.g. SESIP5 with a Secure Element, SESIP3 with a dedicated physical area protected by fault detectors.

Annex A RELATION OF ASSURANCE LEVELS AND ATTACK RATING RELEVANT TO THIS SESIP PROFILE (INFORMATIVE)

The following table gives further information about what SESIP assurance levels and what augmentations are relevant for this SP (first column), how this maps to Common Criteria EAL levels and augmentations (second column). Furthermore, information is added about how this relates to vulnerability analysis and testing levels (AVA_VAN) with corresponding attack potential and attack rating thresholds according to the JHAS methodology and interpretation.

Please note that the table is for informative purposes and depicts the state as of writing this document. The levels are maintained in the SESIP, Common Criteria, and JIL/JHAS standards (see [SESIP]).

Table A-1: Mapping of Assurance Levels and Attack Potential Levels Relevant to this SP (Informative)

SESIP Assurance Levels	Relevant to this SESIP Profile	Assurance	Resistance against Attack Potential	Attack Rating Range
1	no	AVA_VAN.1	basic	16 – 20
2	no	AVA_VAN.2	basic	16 – 20
2^{wb} [8]	yes	AVA_VAN.2 based on source code review	basic	16 – 20
3	yes	AVA_VAN.3	enhanced basic	21 – 24
4	yes (import of CC EAL4+)	AVA_VAN.4	moderate	25 – 30
5	yes (import of CC EAL4+)	AVA_VAN.5	high	31 and above

[8] SESIP 2^{wb} level is not an augmented level: ADV_IMP is not part of the evaluation activities; however, the source code is provided to support the AVA_VAN activity.

Annex B CLARIFICATION ON THE USE CASES RELEVANT TO ATTACK RATING (INFORMATIVE)

B.1 Identification Phase

In the identification phase, it must be assumed that the attacker can easily obtain one or several samples of the TOE for analysis and reverse engineering. Also, additional information such as manuals, firmware images, SDKs, etc., may be publicly available.

An attacker is expected to spend considerable effort to understand and analyze the TOE.

The analysis could also involve physical attacks, such as side channel attacks and fault injection attacks. Also, the attacker could abuse test features and debug interfaces (such as a JTAG interface), and attempt to circumvent access controls, such as by using a debug password.

B.2 Exploitation Phase

In the exploitation phase, the applicability of a particular attack path must be judged based on the use case of the IoT Platform.

Under certain assumptions, such as "trusted user only/trusted code only", some attack paths may be impractical. This is discussed in detail below.

B.3 Use Cases

The SP defines different use cases:

- Assumptions about the trustworthiness of the user
  - **[trusted user only]**
  - **[any user]**
- Assumptions about the trustworthiness of the executed code
  - **[trusted code only]**
  - **[any code]**

B.3.1 Trusted User Only

In this use case, the environment provides physical protections to the TOE and only trusted users (i.e. the legitimate owner different from an attacker) can physically access it. Implicitly, it is assumed that the legitimate owner of the device is trusted, and the device is not left unattended in a public place (e.g. a hotel).

Example: A robotic vacuum cleaner that is used in a private household

In the exploitation phase, the attacker is assumed to have no physical access to victim devices. This renders exploits that require physical access impractical and some attack paths impractical.

Examples:

Access a debug port (such as JTAG) on the PCB, which is not connected.

Fault injections by light impulses or glitches on the hardware.

Side channel attacks requiring a probe on the hardware.

Any exploits that can be mounted by software can be applicable, even if these attacks would be based upon hardware weaknesses.

Example: "Spectre" attack is applicable.

Note that even though the user of the device is trusted, he may inadvertently execute malicious code.

Also, if an exploit can be mounted by accessing an interface that connects the TOE to untrusted devices, the attack would be relevant; the accessed interface is then considered a remote one (this could be covered by assumptions and/or user guideline).

Example: The TOE can be exploited via connection to a USB interface. It could be connected to another device that may be under control of an attacker (e.g. a PC with connection to the Internet).

Another consideration applies to an asset (such as a key) that is not diversified. In this case, the attacker may simply buy a device and obtain the asset during the identification phase, using logical or physical attack methods, allowing the building of a remote attack. In such a case, the exploitation phase is straightforward to be applied to victim devices, and the attack path is applicable to this use case.

Example: The device has a maintenance interface that allows access via ssh. The password is identical for all devices. The attacker learns that password from his own device during the identification phase, using side channel attacks. He can then exploit it remotely on a different device.

B.3.2 Any User

This use case would apply, for example, in cases where the legitimate owner of the device is not trusted.

Example: A digital media player that implements digital rights management (DRM)

It could also apply to a device that is (sometimes or permanently) located in a public place.

Example: A mobile phone that is left unattended in a hotel room ("evil maid" attack)

In this use case, the exploitation phase allows modifications of the TOE, abuse of debug interfaces, or physical attacks such as voltage glitching and side channel attacks.

Note that all attacks considered in **[trusted user only]** are also applicable.

B.3.3 Trusted Code Only

In this use case, only code originated from the vendor is expected to run on the IoT Platform. There is no facility to run third-party code.

Example: A control unit in an industrial appliance

A priori, the attacker is unable to run malicious code on the IoT device containing the TOE. This means he is limited in both the identification and exploitation phases of the attack.

For the identification phase, an attacker would obtain an MCU/MPU development board, or a software emulator. Also, he would attempt to obtain a firmware image from the IoT device, to be analyzed.

As before, the attacker is assumed to own one or several samples of the TOE, which can be used for logical and physical attacks such as glitch attacks, side channel attacks, etc., for the analysis of the device.

For the exploitation phase, the attacker is prevented from executing arbitrary software on the device. However, if the identification phase has revealed vulnerabilities that will allow the loading of untrusted code, this will constitute a valid attack path for this use case.

Example: A buffer overflow in the network stack can be exploited by sending specially crafted IP packets. The vulnerability would also allow execution of malicious code. This constitutes a valid attack path for this use case. However, it is only a partial attack.

If, in addition, the TOE is operated by untrusted users, the attack path may also involve physical attacks, such as glitch attacks, with the goal to bypass access controls that prevent the execution of untrusted code in the exploitation phase.

B.3.4 Any Code

In this use case, code from the user or a third-party entity can be loaded and executed on the IoT Platform. It can also be the case that code developed by the first party is not considered trusted, for instance when it has not been subjected to the same level of security analysis.

For the exploitation phase, we assume the attackers can freely upload their own code and execute it, if this is allowed by the TOE as part of the intended use case, or make use of the existing code.

Example: If the TOE implements a "trusted execution environment" or similar, and any code can be run in the "untrusted" domain outside that environment, then the exploit can involve such code running in the "untrusted" domain.

If the TOE allows upload and execution of third-party code only if the code is signed and the signature verification is in the TOE scope, then the exploit includes the necessary steps to bypass the signature verification or to obtain a valid signature. The effort must be included in the rating.

Note that all attacks considered in **[trusted code]** are also applicable.

B.3.5 Any Code & Any User

Within the considered attack potential, any attack path, logical and physical, is applicable, for both the identification and exploitation phases. Overall rating must remain below the required attack potential for the assumed SESIP level.