GLOBALPLATFORM®
THE STANDARD FOR SECURE DIGITAL SERVICES AND DEVICES

GlobalPlatform Technology

Security Evaluation Standard for IoT Platforms (SESIP)

Version 1.0

Public Release

March 2020

Document Reference: GP_FST_070

THIS SPECIFICATION OR OTHER WORK PRODUCT IS BEING OFFERED WITHOUT ANY WARRANTY WHATSOEVER, AND IN PARTICULAR, ANY WARRANTY OF NON-INFRINGEMENT IS EXPRESSLY DISCLAIMED. ANY IMPLEMENTATION OF THIS SPECIFICATION OR OTHER WORK PRODUCT SHALL BE MADE ENTIRELY AT THE IMPLEMENTER'S OWN RISK, AND NEITHER THE COMPANY, NOR ANY OF ITS MEMBERS OR SUBMITTERS, SHALL HAVE ANY LIABILITY WHATSOEVER TO ANY IMPLEMENTER OR THIRD PARTY FOR ANY DAMAGES OF ANY NATURE WHATSOEVER DIRECTLY OR INDIRECTLY ARISING FROM THE IMPLEMENTATION OF THIS SPECIFICATION OR OTHER WORK PRODUCT.

Contents

Figures

Tables

1 Introduction

This document specifies the general requirements for the competence, impartiality and consistent operation of the Security Functional Requirements and Security Assurance Requirements for the Security Evaluation Standard for IoT Platforms (SESIP).

This document is structured as follows:

- Chapter 2 provides an overview of SESIP presenting the main principles of the evaluation methodology.
- Chapter 3 defines the SESIP Security Functional Requirements, catalog of security requirements for IoT products.
- Chapter 4 describes the SESIP Security Assurance Requirements, describing five hierarchical levels of security assurance for Connected Platforms.
- Annex A – provides a diagram of the overall SESIP functioning described in main chapters.
- Annex B – provides guidance for the attack potential rating methodology.
- Annex C – provides several use-cases as worked examples.
- Annex D – provides a Security Target template.

1.1 Audience

This document is intended primarily for the use of all stakeholders of the security evaluation of Connected Platforms. This document is applicable to all organizations performing SESIP activities.

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 Connected Platforms.

The use of this document will facilitate cooperation between laboratories and other bodies, and assist in the exchange of information and experience, and in the harmonization of standards and procedures. 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

Table 1-1: Normative References

Standard / Specification	Description	Ref
ISO/IEC GUIDE 99:2007	International vocabulary of metrology — Basic and general concepts and associated terms (VIM)	[ISO Guide 99]
ISO/IEC 15408-3:2008	Information technology – Security Techniques – Evaluation criteria for IT security – Part 3: Security assurance components	[ISO 15408-3]
ISO/IEC 17000:2004	Conformity assessment – Vocabulary and general principles	[ISO 17000]
ISO/IEC 17065:2012	Conformity assessment – Requirements for bodies certifying products, processes and services, September 2012	[ISO 17065]
Common Criteria Part 1, 2017	Common Criteria for Information Technology Security Evaluation – Part 1: Introduction and General Model. CCMB-2017-04-001, version 3.1, revision 5, April 2017	[CC Part 1]
Common Criteria Part 2, 2017	Common Criteria for Information Technology Security Evaluation – Part 2: Security functional components. CCMB-2017-04-002, version 3.1, revision 5, April 2017	[CC Part 2]
Common Criteria Part 3, 2017	Common Criteria for Information Technology Security Evaluation – Part 3: Security assurance components. CCMB-2017-04-003, version 3.1, revision 5, April 2017	[CC Part 3]
Security IC Protection Profile, PP-0084, 2014	Security IC Platform Protection Profile with Augmentation Packages. Version 1.0, 2014. Certified by BSI under reference BSI-CC-PP-0084-2014.	[PP-0084]
cPP for Network Devices, v2.1	collaborative Protection Profile for Network Devices. Version 2.1, September 2018	[ND cPP]

Table 1-2: Informative References

Standard / Specification	Description	Ref
Attack Potential for Smart Cards, latest version	Application of Attack Potential for Smart Cards	[APSC]
PSA Certified Level 2 Lightweight Protection Profile, 2019	PSA Certified™ Lightweight Protection Profile, JSADEN0002, Version Beta 02, February 2019	[PSA2 PP]
DTSec Protection Profile, 2017	DTSec Protection Profile for Connected Diabetes Devices (CDD). Version 2.0, November 2017	[DTSec PP]

1.4 Terminology and Definitions

Selected terms used in this document are included in Table 1-3. Additional terms are defined in [ISO 17000].

Table 1-3: Terminology and Definitions

Term	Definition
Certification body (CB)	Throughout this document the term “certification body” is used in keeping with the terminology of [ISO 17065], and when used the term holds the same meaning as “conformity assessment body” as defined in [ISO 17000].
Certification scheme	Certification system related to specified products to which the same specified requirements, specific rules and procedures apply [ISO 17000]. A scheme may be developed among others by a certification body or by a “scheme owner” representing a specific group of interests. The scheme may contain requirements on conformity assessment procedures and functions of the certification bodies complementary to those established by [ISO 17065].
Conformity assessment	Demonstration that specified requirements relating to a product, process, service, person, system or body are fulfilled.
Connected Application	Software developed by an IoT vendor, implementing IoT end-user use case based on the underlying Connected Platform. May be referred to as “Application” when there is no ambiguity.
Connected Platform	Combination of hardware and software that provides a runtime environment for a Connected Application. A Connected Platform implements security features and makes security services available to the Connected Application. May be referred to as “platform” when there is no ambiguity.
Connected Platform developer	Developers who build platform (parts) and supply these to product vendors or to other platform vendors, who need to certify the security of the platform (parts) that they build. May be referred as “platform developer” when there is no ambiguity.
Connected Platform part	Hardware and/or software that implements a subset of the features of a Connected Platform. It can be developed and evaluated separately, for example the hardware, a cryptographic library, an OS. May be referred as “platform part” or “part” when there is no ambiguity.
Connected Product	Combination of a Connected Platform and a Connected Application that a product vendor puts on the market. May be referred as “product” when there is no ambiguity.
KeyStore	Repository, typically a file, in which certificates, private keys or secrets can be stored.
Platform evaluator and certifier	Parties in a scheme who verify whether a Connected Platform is secure in accordance with the standard described in this document. May be referred as “evaluator” / “developer” when there is no ambiguity.

Term	Definition
Product vendor	Vendors of complete IoT systems that include Connected Products based on the certified Connected Platform (parts). Product vendors assemble Connected Platform (parts) and Applications into Connected Products, and they deliver these products to the users.
Recommendation	Expression in the content of a document conveying a suggested possible choice or course of action deemed to be particularly suitable without necessarily mentioning or excluding others.
Requirement	Expression in the content of a document conveying objectively verifiable criteria to be fulfilled and from which no deviation is permitted if compliance with the document is to be claimed.
User	External entity, human or IT, interacting with the Connected Platform.

1.5 Abbreviations and Notations

Table 1-4: Abbreviations and Notations

Abbreviation / Notation	Meaning
CB	Certification body
CC	Common Criteria
DTSec	Diabetes Technology Society
PSA	Platform Security Architecture
RoT	Root of Trust
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
ST	Security Target
TOE	Target Of Evaluation

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 precisions; all non-trivial changes are indicated, often with revision marks.

Table 1-5: Revision History

Date	Version	Description
March 2020	1.0	Public Release

2 Overview

The present chapter provides an overview of the essential principles underlying SESIP:

- A strong formalism as a basis of the methodology.
- A threat model adapted to the IoT ecosystem.
- A life cycle adapted to Connected Products in the IoT ecosystem.
- The reusability, an essential objective of SESIP, in order to handle at an acceptable cost the increasing complexity of the Connected Platforms that need to be evaluated in the IoT ecosystem.
- The accessibility, which is required to encourage product vendors to leverage the security features included in evaluated Connected Platforms; the results of an evaluation must be accessible and exploitable by security-proficient developers without the need to be evaluation specialists.
- The security self-assessment in SESIP.

2.1 A Strong Formalism

SESIP is based on the Common Criteria methodology ([ISO 15408-3]) specialized for the evaluation of Connected Platforms in the context of IoT. The Common Criteria foundation provides the formalism, while the specialization for a specific set of security products allows the optimization of the evaluation process.

SESIP can then be seen as a variant of the Common Criteria framework, from which it adopts many guiding principles:

- SESIP follows the main Common Criteria principles as defined in [CC Part 1].
- SESIP is not using the SFR catalogue defined in [CC Part 2] but keeps the concept of a catalogue of SFRs, specialized for the IoT ecosystem. Also, each SFR targets a full security purpose rather than of being split in low level mechanisms for genericity purpose.
- SESIP is using the SAR catalogue as defined in [CC Part 3], with some refinements of the SARs defined in [CC Part 3] and addition of new ones. Also, SESIP is not using “EAL” packages defined in [CC Part 3], but defines its own assurance packages (SESIP levels) adapted to IoT ecosystem: the SESIP levels.

SESIP, like Common Criteria, is an evaluation methodology, which defines as precisely as possible how to evaluate the security of a product, here a Connected Platform. Similarly, SESIP does not define any specific procedure, nor does it organize explicitly the mutual recognition principles between certificates, and only provides guidance and directions. A SESIP certification scheme based on this SESIP evaluation methodology must be defined in another document, by the certification scheme owner.

2.2 IoT Use Cases and Threat Model

IoT is a broad term, but always contains a product (“thing”) and some form of connectivity (“internet”). SESIP focuses on the “thing” side of IoT, and on the security of Connected Products based on Connected Platforms.

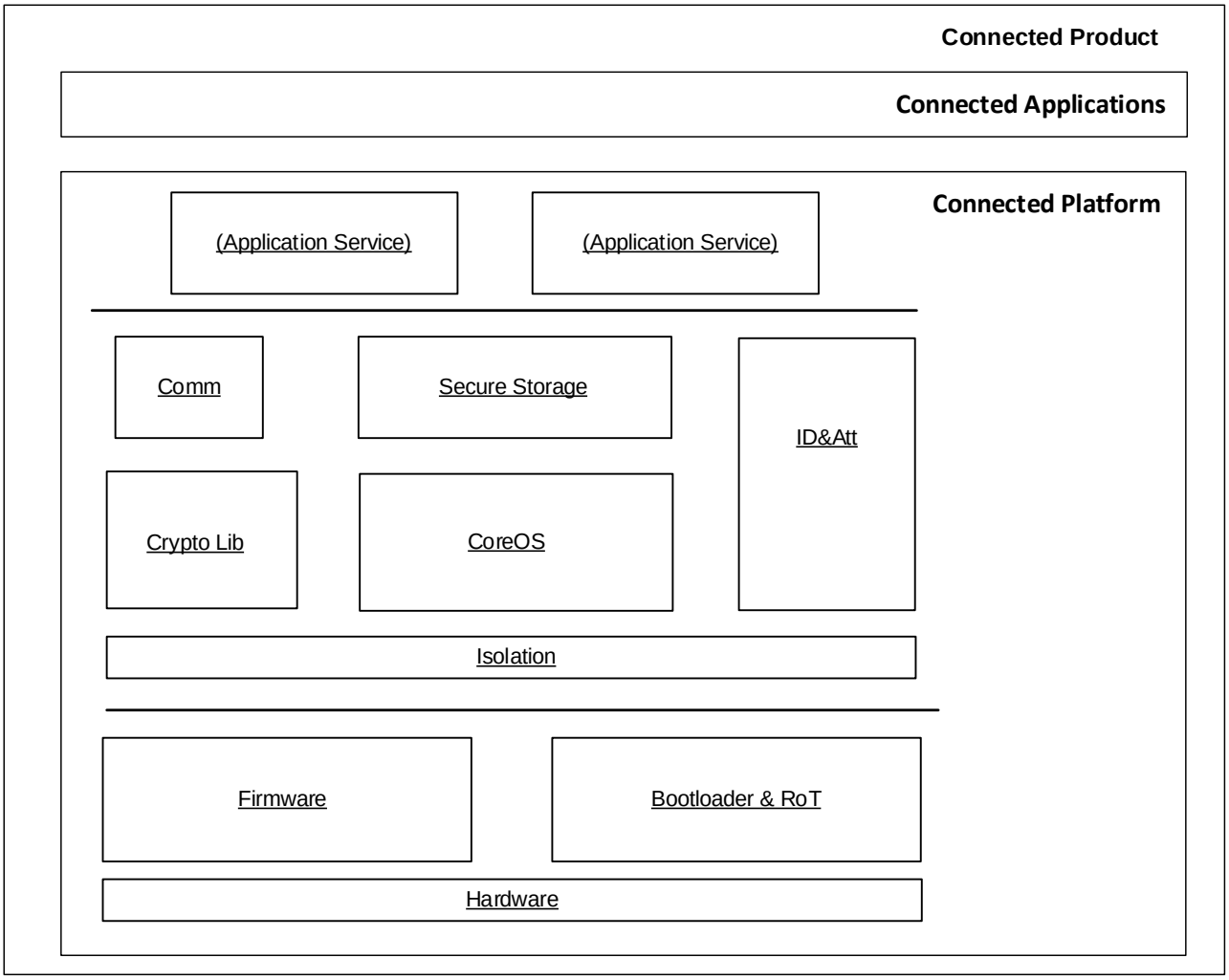
Architecture

A Connected Platform typically includes the following components:

- hardware (processing unit, memory, possibly a secure element, at least one network interface, possibly some sensors);
- an operating system, providing a foundation to run Connected Applications on the hardware;
- a network connectivity layer, allowing connecting the product to backend or other products;
- software application services offered to Connected Applications, providing an application framework to product vendors.

It is assumed that the Connected Platform includes at least one network interface that is directly or indirectly connected to a network and exposed to potential attackers.

Figure 2-1: Example of Connected Product Architecture



Assets

The main assets of a Connected Platform are:

- User data (local): Privacy concerns are essential, and protections of integrity, authenticity, and confidentiality must be provided.
- User data (authentication data): Confidentiality is required for secrets, and secondary data (like counters) must be appropriately protected (integrity, confidentiality).
- Data in transit (internet): Confidentiality and integrity are often essential, as well as authenticity and authentication of the other party.
- Data in transit (local): Integrity is often essential as well, confidentiality is not a systematic requirement, and authenticity and authentication of the other party are less common.
- Code, including platform code and application code: Integrity and authenticity are strong requirements; confidentiality is optional.
- Product identity: Integrity and unicity are required.
- Configuration and system data: Integrity and authenticity are required.
- Life cycle related data: Integrity is required.

It is understood that the most limited Connected Platforms do not provide a complete coverage, but such limitations must be carefully motivated when claiming SESIP SFRs (e.g. limited bandwidth, legacy protocol).

The assets may be further categorized into different criticality levels that will be protected at the appropriate level in the platform (part) - in the case of a multi-assurance platform, see section 2.4.2. For instance, there may be different levels of cryptographic keys, depending on their function and life cycle. In that case:

- protection mechanisms must be appropriate at every level;
- assets must be usable without disclosing them or otherwise to a lower level;
- usage of the assets from a lower level must be appropriately controlled (access control).

Attackers and threats

Base scenario: The minimum and mandatory threat model in SESIP is an attacker with only remote (no physical) access to the Connected Platform during the exploitation phase (see Annex B). This addresses the main IoT concern of a scalable attack exploited using a remote connection to the Connected Platform.

Nevertheless, the attacker can perform any type of preliminary attacks on a Connected Platform (part) he owned, including physical attacks; this then must be considered for the base scenario in an identification phase (see Annex B).

Also, in this base scenario, threats related to untrusted software that could be loaded onto Connected Platform are not considered.

Extended scenario – physical access: When Connected Platforms are physically accessible to attackers, the threat model can be expanded and covered by the use of the SFRs “Limited Physical Attacker Resistance”, “Physical Attacker Resistance”. The typical example scenarios where attackers have a physical access to a victim product are:

- Connected Platform deployed outside of physical protected environment; e.g. doorbell, outside IP camera.
- Temporarily physically accessed; e.g. “evil maid” attacks where the attacker has temporary physical access to the product while already acquired by an end-user, or “supply chain” attacks where the attacker delivers a compromised product to the target.

Extended scenario – untrusted software: When untrusted software can be loaded onto Connected Platforms, either by the end-user or by an external entity, and that could have an impact on this platform, its parts or applications, the base threat model can be expanded and covered by the use of the SFRs “Software Attacker Resistance: Isolation of Platform”, “Software Attacker Resistance: Isolation of Platform Parts”, and “Software Attacker Resistance: Isolation of Application Parts”.

2.3 Connected Product Life Cycle

Different life cycle models can be applied to Connected Products, and to the Connected Platform that compose each product. Nevertheless, some patterns can be found in most products which are significant for security:

Vendor provisioning is the phase during which the product is provisioned with credentials that are shared with the vendor's backend, and that allow the product to communicate securely with the backend and to perform management operations. This phase typically concludes with the delivery of the product to the customer.

User provisioning is the phase during which the product is provisioned with a user's credentials and specific data that allow the product to represent that user. This phase typically concludes with the normal usage phase of the product.

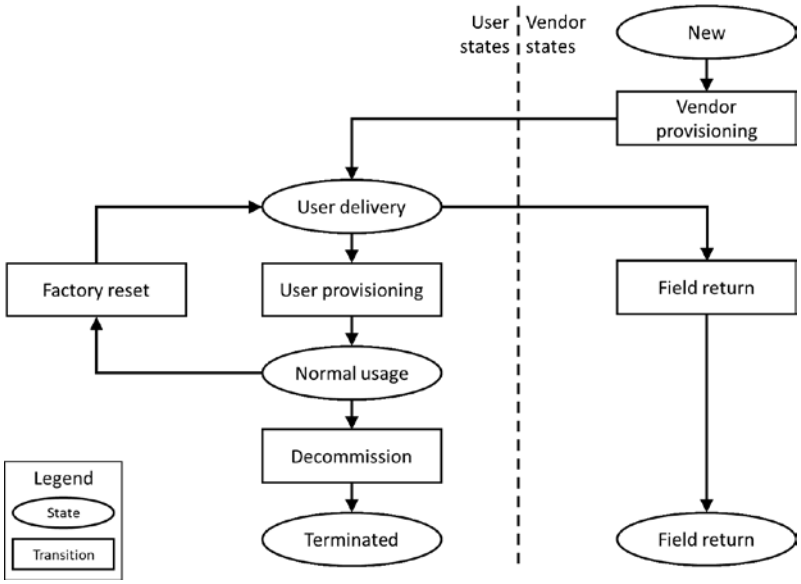
Normal usage is supposed to be the product's normal state, until one of the following events occurs:

- The user applies a factory reset, which removes all user-related data and credentials, and prepares the product to be transferred to another entity (e.g. for resale, for return, or even for temporary storage). The product is then ready again for user provisioning, but a user should not have the ability to return the product to an earlier life cycle phase.
- The user terminates the product, before discarding it. This **Terminated** state is irreversible.

Some products may include an additional state related to **Field return**, during which specific debugging features may be available. All user data and credentials shall have been removed before reaching that state.

The product life cycle shown in the figure hereafter is used as a reference in the SFRs when references to a life cycle are required:

Figure 2-2: Reference Product Life Cycle



Note that the vendor states are only reachable by the vendor, either before delivery of the product, or after return of the product by the user.

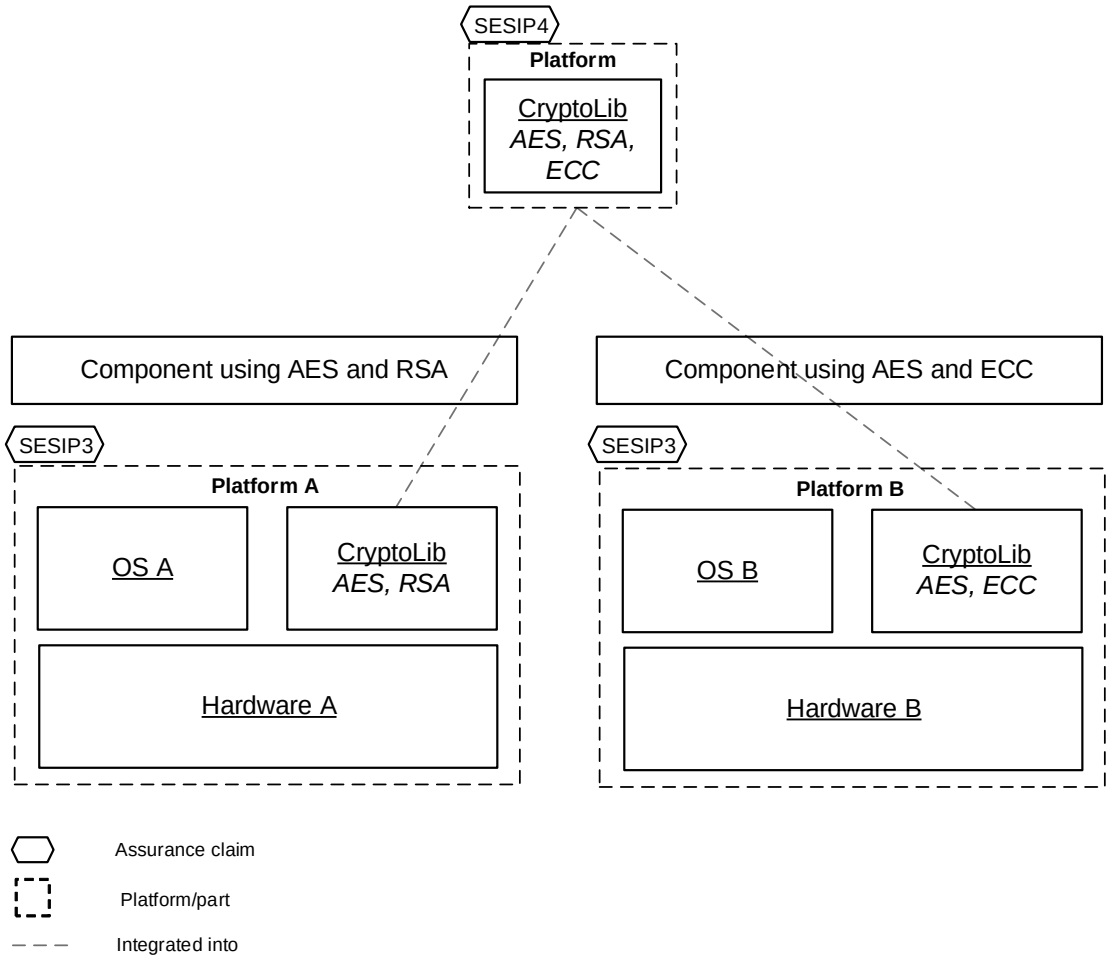
In addition to the product life cycle, the Connected Platform and some of its parts may have different life cycles that are also significant to security. Such security life cycles are product-specific, and their contribution to the Connected Platform (part)'s security should be described in the corresponding Security Target.

2.4 Reusability in SESIP

Connected Products are complex, often much more than most of the products that have had their security formally certified until today. SESIP recognizes this by providing a dedicated methodology for the Connected Platforms on which these products are based. Connected Platforms are often built by assembling several pre-existing hardware and software components; some of them include security components that protect critical assets and need to be evaluated at high assurance level. Such components are often integrated in several Connected Platforms targeting different use cases.

SESIP methodology defines ways to independently evaluate subset of components, then called platform parts, and reuse the evaluation results in any Connected Platform. Those results can come from an evaluation under SESIP methodology, but also from other compatible external evaluations; in particular, for the highest assurance levels, Common Criteria results can be reused.

Figure 2-3: Example of Evaluation Results Reuse in Several Platforms Offering Different Services



2.4.1 Building Connected Products from Connected Platforms

Reuse of external evaluations

As mentioned in the previous section, SESIP focuses on the "Things" in IoT, and more specifically on the solutions on which these connected things are built, which we call Connected Products. Every Connected Product belongs to a category or a vertical, and dedicated security standards are likely to be built for the most common types of Connected Products (e.g. Consumer IoT, Industrial IoT, Connected Vehicles, etc.); for each type, a specific risk analysis is needed to determine the appropriate functional and assurance requirements, and may then results in a specific evaluation scheme creation.

In such multi-scheme context, SESIP security requirements are defined in a way enabling the establishment of equivalence with other schemes requirements, to then perform compatibility analysis; this allows the reuse of evaluation results between schemes.

Reuse of platform parts evaluations

A typical Connected Platform is not a monolithic component, as it comprises some hardware, including at least one [micro]controller or [micro]processor, and some software, including at least an operating system. A Connected Platform may include many more components, for instance related to communication or security. A vendor typically builds its Connected Platform by selecting hardware and software components, most likely from different third-party vendors, and then assembling them.

Security evaluation assessment needs to be part of the delivery from every vendor in that supply chain, ending with the integrator in charge of ensuring that the full Connected Platform is secure. To maintain security through the whole assembly process can be quite complex, unless all the vendors of those components (hardware and software) use a common methodology.

In order to address this, SESIP allows the evaluation of platform parts (set of components, for instance defined by developer or type of feature), individually or in composition (see next section), in such a way that those platform parts evaluation results remain applicable in different Connected Products. This is in particular based on security integration guidance for each component; then, the assessment of Connected Platforms integrating evaluated platform parts only requires the verification that secure integration guidelines and composition rules (see next section) are respected.

Using SESIP as core methodology allows vendors having a clear understanding on the assumptions and guidance of the third-party components they integrate, by reusing evidences provided by the components' vendors during the evaluation of their platform (part).

Reuse of platform evaluation in Connected Products

Most of the Connected Products will be built by vendors by developing one or several dedicated Application(s) on top of a generic or off-the-shelf Connected Platform. SESIP focuses on the Connected Platform to provide a foundation for the security assessment, and potentially the certification, of Connected Products; it addresses the security level of the whole operating system including services used for the storage, installation, initialization and execution of this Application as well as its data protection.

2.4.2 Additive Composition within SESIP

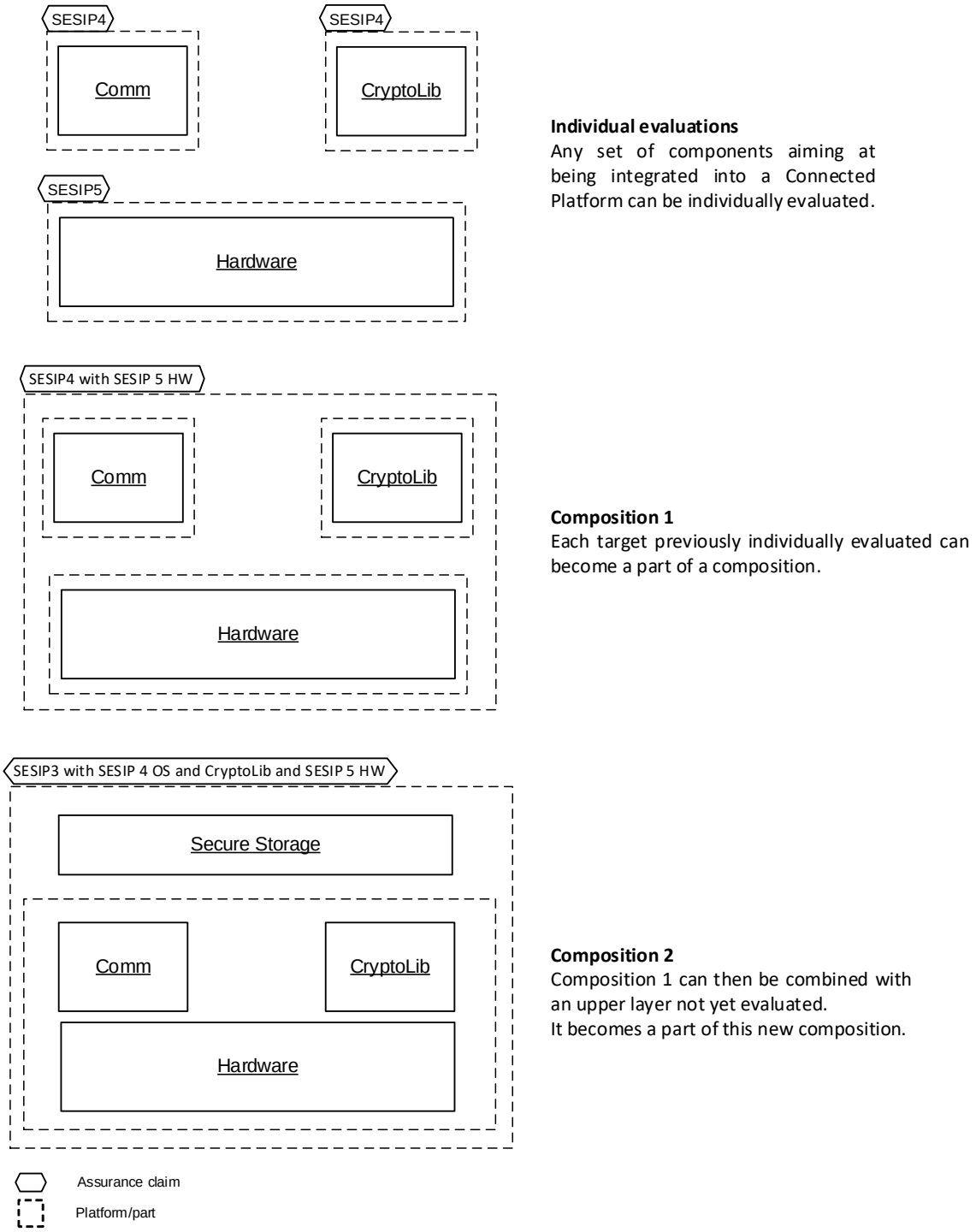
The additive composition objective is to facilitate the evaluation of a Connected Platform reusing previous evaluations results of platform (parts).

The overall target of an evaluation is always called platform; in case of a composition, previously evaluated parts and the part(s) to be evaluated are called platform parts.

SESIP allows different composition scenarios:

- Combination of already evaluated platform parts;
- Combination of platform part(s) to be evaluated with already evaluated platform part(s);
- Combination of a platform to be evaluated with already evaluated platform parts.

Figure 2-4: Example of Compositions Scenarios



Composition evaluation rules

Individual evaluation of a platform part must produce composition guidelines listing rules to be respected by any other platform part interacting with it. This must be described as an objective for the environment and referred to specifically in the guidance as per SESIP ASE and AGD requirements (see details in section 4.1.3).

Evaluation of platform parts composition must assess that the guidance (including objectives for the environment, user guidance, integration guidance, etc.) of each platform part is respected.

Evaluation of platform parts composition must assess the impact of the composition on the correct security functioning of the platform parts.

Assurance level of compositions

Composition may occur between platform parts that are evaluated at different assurance levels. By default, the composition can claim at most the lowest assurance level of the platform parts it is composed of. For instance, a Connected Platform composed of a SESIP2 platform part and a SESIP5 platform part, cannot be certified at an overall level higher than SESIP2 without providing additional evidence about the SESIP2 part, or its usage by SESIP5 platform part.

Nevertheless, in such a case, the Security Target may identify a subset of the SFRs, requiring higher assurance. In that case, the Security Target can claim the high-assurance part provided the SFRs covered by the higher part are clearly identified to the reader. Using the same example as above, the Connected Platform could claim being at SESIP2 with a SESIP5 part covering only the SFRs of the SESIP5 part. The assurance level for such an evaluation shall be referred to as “SESIP2 with SESIP5 part(s)”.

The SESIP level of a platform part individually could also be increased once composed with another platform part providing security features with a higher security level.

Such multi-assurance evaluations are expected to fulfill industry requirements for composition. For instance, industry requirements may mandate that critical assets such as some cryptographic keys or credentials be specifically protected using a tamper-resistant product like a secure element while other assets less critical will require lower level of protection. In that case, the Connected Platform is required to have a subset evaluated to a higher assurance level, as described above.

2.5 Accessibility and Transparency

IoT security remains a challenge today. A SESIP evaluation will help IoT vendors to select Connected Platforms that provide an appropriate level of security, but SESIP must also ensure that these vendors have the ability to properly leverage in their products the security features provided by evaluated platforms (parts). This is achieved mostly through the Security Target, which must become a document exploitable by developers who are proficient in security but are not evaluation specialists. This can be achieved by ensuring that Security Targets are transparent and readable, that the security choices are explicit and motivated, and that SESIP can ensure some consistency between Security Targets.

In that purpose, SESIP Security Targets must respect several key characteristics:

- The security objectives are the concretization of the risk analysis informing the reader about the security features to be covered by the product. They must be described by the Security Target writer such that the developers can leverage these features, and to facilitate the reuse of SESIP evidences.
- The SESIP Security Functional Requirements (SFRs) are tailored for Connected Platforms, which are the expression of the security objectives into implementation requirements. In SESIP, each of them is covering a full security purpose by itself. This allows an intuitive understanding of the security requirements by the Security Target users. Note that flexibility for requirements construction is lower than when purposes are met by several SFRs, however such flexibility has been found not beneficial in a specific context where needs are predictable, as for Connected Platform. SESIP pre-defines a set of such SFRs presented in the next chapter.
- The specification summary written by the platform evaluator must be at a sufficient level of details to demonstrate that SFRs are met by the specific implementation.
- The summary description of flaw remediation process by the platform (part) developer, explaining in particular how the discovery of a vulnerability and the subsequent analysis may lead to the development and distribution of a platform update or to other corrective measures.
- The reference to user guidance and the description how a prospective user can access this documentation.

Note that accessibility does not necessarily imply oversimplification; in particular, the semantics of SESIP SFRs is precisely defined, so there is no ambiguity about the meaning of the SFR, even expressed in plain language.

The SESIP Security Target and the documentation that it refers to is intended to be publicly accessible documentation, or at least to prospective adopters of the platform (part); this can be refined in individual schemes, depending on the security sensitivity of the evaluated platforms (parts).

2.6 Security Self-assessment in SESIP

Security self-assessment is achievable with SESIP, but the self-assessment has many different meanings in the certification industry, it is therefore important to define it here.

The security self-assessment by the developer is based on the demonstration that all security requirements are properly covered in the form of a rationale included in the SESIP Security Target.

Such rationale must be provided for:

- every Security Functional Requirement claimed, describing how this requirement is covered by the implementation;
- in the case of a composition, every Security Objective for the Environment defined by a platform part, to demonstrate how this objective is met and/or define it again as a Security Objective for the platform Environment.
- the flaw remediation claim, providing description of a compliant process.
- each document required by the claimed Security Assurance Level, explaining how that documentation can be accessed.
- publicly known vulnerabilities against the platform (part), providing a vulnerability review.

Every rationale is based on a self-assessment by the developer which must be validated by a third-party platform evaluator, the extent of work performed by the platform evaluator depends on the selected assurance level.

2.7 Catalog of Security Features and Assurance Packages

SESIP includes a catalog of Security Functional Requirements (SFRs) as an essential part of the methodology, which allows for a consistent definition of platforms (parts) and supports a fair comparison between those. This catalog defines a set of security features that are essential to platforms (parts), and for which there is a shared understanding in the community. These security features are therefore more likely to be accessible to product vendors; they also simplify reusability and participate to SESIP's formalism (see previous sections).

This set of features will evolve over time, following the evolution of IoT security challenges, the growing usage of SESIP methodology, and the security responses brought by Connected Platforms. Platform (part) vendors may also want to differentiate their offer by including platform-specific or innovative security features that bring added-value to their customers, and it is possible to have such features evaluated following the SESIP methodology.

However, it must also be ensured that the methodology provides the stability required to build and maintain a coherent certification ecosystem. Therefore, in the case where a feature does not match any of the SFRs defined in the SESIP catalog, then a vendor may be allowed by the scheme to add product-specific SFRs, as long as they are explicitly identified as such, and clearly separated from the SFRs already in the catalog.

Regarding Security Assurance Requirements (SARs), vendors are not allowed to claim any additional requirement in a SESIP evaluation. However, a specific certification scheme may include limited refinement of the SARs in the SESIP levels.

3 Security Functional Requirements

Connected Platforms need to provide developers and evaluators of IoT applications with functionality to build secure products and allow efficient verification of the accurate use of these secure platforms by users, developers and evaluators/certifiers. Such needs are expressed by the Security Functional Requirements (SFRs) which are generic security features to be implemented by Connected Platforms; those SFRs have been defined subsequently to an initial risk analysis dedicated to the IoT ecosystem.

The SFRs are described with the wording of the requirement, the “value”, and the “considerations”.

- The wording of the requirement describes what must implemented.
- The “value” section describes what is added in value by a platform providing this functionality, to the users, evaluators and developers of composite IoT products and platforms.
- The “considerations” section describes what aspects should be considered in the evaluation and certification of this security functional requirement.

Additionally, the “Traditional CC” section describes how the optimized CC encoding in SESIP and the encoding in traditional CC could be related, allowing translation between the two.

Important note

All Security Targets must include the “Verification of Platform Identity” SFR, and all must either include the “Secure Update of Platform” SFR or argue under ALC_FLR.2 why updates are not applicable.

SFR identification is done by using the subsection title, e.g. “Verification of Platform Identity” (not the section number as this may change in future document releases).

3.1 Identification and Attestation of Platforms and Applications

Identification and attestation of applications, platform and platform parts allow customers, developers and evaluators to verify they have the evaluated product. More complex attestation allows for a wider scope of what is attested, and higher assurance on that which is attested. The SFRs in this subsection are to be used to express identification and attestation requirements.

3.1.1 Verification of Platform Identity

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

Value

Users can only verify they have a secure product, if they can obtain the identifications of the product parts (application and Connected Platform). The platform is a crucial building block of that identification process.

Evaluators and developers of composites need to be able to verify the identity too (but might require attestation for higher assurance, see section 3.1.3).

Considerations

The ST describes the platform, thereby defining all its parts.

Developers of platform (parts) are required to keep the identification (globally) unique: All products from that developer that identify in the way the certified product is identified, shall be the certified product. This functional requirement is mandatory for all platforms, to ensure customers can verify that they have the correct certified platform. This includes any other composition of platform parts.

Traditional CC

[CC Part 2] defines the requirement for identification of the TOE distributed over ASE_INT, AGD_PRE, and the identification tasks in ALC_CMC, ATE and AVA. SESIP centralizes this to a function in the platform to enable automated checking of the platform identification, allowing for more efficient compositions checks and future compliance checks.

3.1.2 Verification of Platform Instance Identity

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

Value

Developers of composites may need a unique identifier for a specific instantiation of a platform.

Considerations

Developers of platform (parts) are required to keep the individual instantiation identification (globally) unique in combination with the “Verification of Platform Identity”.

A typical example of platform identify is its serial number of a platform.

Traditional CC

[CC Part 2] does not define an equivalent security requirement. [PP-0084] has the SFR FAU_SAS.1 with an open operation allowing encoding as such:

FAU_SAS.1 Audit storage

FAU_SAS.1 The TSF shall provide the test process before TOE Delivery with the capability to store [*selection: the Initialization Data, Pre-personalization Data*], and a unique identification of the specific instantiation of the TOE in the [*assignment: type of persistent memory*].

3.1.3 Attestation of Platform Genuineness

The platform provides an attestation of the "Verification of Platform Identity" and "Verification of Platform Instance Identity", in a way that cannot be cloned or changed without detection.

Value

Users, developers and evaluators can verify that they have the genuine (secure) product, not an insecure/incomplete clone. This gives more assurance to those parties and makes the genuine secure product more distinctive and valuable. Together with the "Verification of Platform Instance Identity" SFR, authentication of this platform can be shown.

Users of the platform certificates issued against this standard, such as dedicated schemes for products, can use this attestation as their way of identifying the genuine product.

Considerations

The genuine identification is used to ensure a given platform is genuine instantiation of the platform and not a product posturing as one (clone). Hence the mechanism and its keys are valuable to an attacker seeking to clone the platform or otherwise misuse the developer's brand and reputation. As the developer is potentially held accountable for clones identifying as the genuine platform, or at least suffers reputation damage for them, such an attacker goal shall be considered as part of the evaluation.

Traditional CC

[CC Part 2] does not define a singular way to encode this. Commonly this is a challenge-response protocol encoded with FCS_COP.

3.1.4 Secure Initialization of Platform

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

Value

Users, developers and evaluators can trust that the platform verified its authenticity and integrity at start-up, hence an operational product is running on a secure platform.

Considerations

A platform detecting a breach of authenticity or integrity may offer "Factory Reset of Platform", "Secure Update of Platform", or "Decommission of Platform" functionality to recover a given product.

Traditional CC

[CC Part 2] does not define an equivalent security functional requirement to encode this. The notion of secure initialization is considered under ADV_ARC.1. However, that is not prescriptive about ensuring authenticity and integrity during the platform initialization, leaving the definition of what is considered to be secure initialization up to the developer to define. SESIP defines explicitly that the authenticity and integrity is checked.

3.1.5 Attestation of Platform State

The platform provides an attestation of the state of the platform, such that it can be determined that the platform is in a known state.

Value

Users, evaluators and developers of composites can verify that the platform is in the evaluated known state.

Consideration

Implies that “Attestation of Platform Genuineness” and “Secure Initialization of Platform” are also implemented.

Traditional CC

[CC Part 2] does not have an equivalent security functional requirement to encode this. Commonly this is left as an item to be addressed through manual checks as described in guidance documentation and considered under AGD_PRE.1, and possibly also AGD_OPE.1.

3.1.6 Attestation of Application Genuineness

The platform provides an attestation of the application, in a way that cannot be cloned or changed without detection.

Value

Users can determine that they have the genuine (secure) product by verifying the application, not an insecure/incomplete clone. This gives more assurance to the user and makes the genuine (secure) product more distinctive and valuable.

Considerations

Implies that “Attestation of Platform Genuineness” and “Secure Initialization of Platform” are also implemented.

Traditional CC

[CC Part 2] does not have an equivalent security functional requirement to encode this. Commonly this is left as an item to be addressed through manual checks as described in guidance documentation and considered under AGD_PRE.1, and possibly also AGD_OPE.1.

3.1.7 Attestation of Application State

The platform provides an attestation of state of the application.

Value

Users, evaluators and developers of composites can verify that the application is in a specific state. This specific state could then be compared to the evaluated secure state.

Considerations

The attested state of the application is here limited to information that is available to the platform, such as the application’s static code and configuration, and other platform-managed information, e.g. an application life cycle state.

Implies that “Attestation of Application Genuineness” and “Attestation of Platform State” are also implemented.

Traditional CC

[CC Part 2] does not have an equivalent security functional requirement to encode this. Commonly this is left as an item to be addressed through manual checks as described in guidance documentation and considered under AGD_PRE.1, and possibly also AGD_OPE.1.

3.2 Product Life Cycle: Factory Reset / Install / Update / Decommission

The platform shall always maintain the security functional requirements claimed, including the boot and shutdown stages.

The SFRs in this subsection are to be used to express other common life cycle steps such as secure installation, update and decommission of the platform and application.

The confidentiality of the platform or application may be important, for example to protect intellectual property rights or because this confidentiality has been assumed during a vulnerability analysis. As these life cycle steps may happen in the field, confidentiality needs to be maintained.

3.2.1 Factory Reset of Platform

The platform can be reset to the state in which it exists when the composite product embedding the platform is delivered to the user, before any personal user data, user credentials, or user configuration is present on the platform.

Value

The user invokes this functionality prior to disposing of the product instance or otherwise potentially allowing an attacker physical access to the product instance, such as reselling it. The user's data is guaranteed to be destroyed, independently of the application running on the platform.

Considerations

The platform shall still be functional after the factory reset, allowing the user to initialize the composite product embedding the platform.

This reset shall destroy all data (including the data of the application) received after user delivery, such that none of this data can be compromised even when the product is also physically accessible to an attacker.

This functionality still allows storage of platform instance unique data, such as data needed for "Attestation of Platform Genuineness" and "Attestation of Platform State", allowing the platform and product to be operational still.

This functionality is not intended to counter attacks where an attacker has temporary physical access and then returns it to the user, those are countered by "Physical Attacker Resistance".

If the application is (partially) defective, it shall still be possible to factory reset the product and then throw the product away, without the (user) data being recoverable in the product still. Platforms with functionality from "Secure Encrypted Storage", "Residual Information Purging", or "Cryptographic KeyStore" typically will be able to fulfil this requirement easier.

The destruction method shall be appropriate for the persistent memory technology and attack potential. See also "Residual Information Purging".

Traditional CC

[CC Part 2] defines a requirement for residual information protection (FDP_RIP.1) that is intended to provide functionality to ensure content of a resource is made unavailable either upon allocation or deallocation of the resource to an object. However, that does not specifically address the scenario where an action is invoked (factory reset) by the user nor the enforcement of active data destruction.

3.2.2 Secure Install of Application

The application can be installed in the field such that the integrity, authenticity *<and confidentiality>* of the application is maintained.

Value

An application may be installed in the field, including at the final end-user and at unsecured product production sites. The composite developers and evaluators can be ensured that the application is not modified (or optionally: disclosed) during this installation.

Consideration

The installation mechanism shall ensure that an application is compatible with the underlying platform in its current version before installing the application.

A platform offering this SFR should consider also offering “Secure Update of Application” as a complement to the “Secure Update of Platform”, allowing the product vendor to provide flaw remediation procedures that cover the entire product.

Traditional CC

[CC Part 2] does not have an equivalent security functional requirement to encode this. Commonly this is left as an item to be addressed through manual configuration of the application in accordance to Preparative Guidance, as considered under AGD_PRE.1.

3.2.3 Secure Update of Platform

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

Value

Addressing security flaws, functional bugs or improvements, may require an update of the platform in the field. The update mechanism should not in itself enable an attack.

Considerations

Note that absence of this functionality shall be explained in the ST under ALC_FLR.2.

The update mechanism shall counter against downgrade attacks (“updating” to an older, potentially more vulnerable version). What is an “older” or “newer” version is defined in ALC_FLR.2.

Note also that updates of the platform shall have a (globally) unique identifier as per “Verification of Platform Identity” and may require their own (re-)evaluation and (re-)certification.

Traditional CC

[CC Part 2] does not have an equivalent security functional requirement to encode this. The notion of secure update is typically considered across a number of security functional requirements, such as trusted channel, FTP_ITC.1, together with the ALC_FLR assurance requirements for timely updates (although that does not address the updating of a deployed instance of the TOE).

A number of collaborative Protection Profiles (e.g. [ND cPP]) include extended components, such as FTP_TUD_EXT.1 to specify requirements for administrator-initiated update, with open operations:

FPT_TUD_EXT.1.1 The TSF shall provide Security Administrators the ability to query the currently executing version of the TOE firmware/software and [*selection: the most recently installed version of the TOE firmware/software; no other TOE firmware/software version*].

FPT_TUD_EXT.1.2 The TSF shall provide Security Administrators the ability to manually initiate updates to TOE firmware/software and [*selection: support automatic checking for updates, support automatic updates, no other update mechanism*].

FPT_TUD_EXT.1.3 The TSF shall provide means to authenticate firmware/software updates to the TOE using a [*selection: digital signature mechanism, published hash*] prior to installing those updates.

3.2.4 Secure Update of Application

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

Value

Addressed security flaws, functional bugs or improvements may require an update of the application in the field. The composite developers and evaluators can be ensured that the application is not modified or disclosed during this update.

Consideration

The update mechanism shall counter against downgrade attacks ("updating" to an older, potentially more vulnerable version). What is an "older" or "newer" version is defined in ALC_FLR.2.

The update mechanism shall ensure that the new version of the application is compatible with the underlying platform in its current version before updating the application.

A platform offering this may consider also offering "Secure Install of Application".

Traditional CC

As stated in "Secure Update of Platform", [CC Part 2] does not have an equivalent security functional requirement to encode this. The notion of secure update is typically considered across a number of security functional requirements (such as trusted channel, FTP_ITC.1, and extended components, such as FTP_TUD_EXT.1 as used in a number of collaborative Protection Profiles) together with the ALC_FLR assurance requirements for timely updates (although that does not address the updating of a deployed instance of the TOE).

3.2.5 Secure Uninstall of Application

The application can be uninstalled in the field such that all application data *<with the exception of <list of objects not destroyed>>* is destroyed.

Value

The user invokes this functionality prior to disposing of the application instance or otherwise potentially allowing an attacker physical access to the application instance. As all application data is destroyed, the application user's data is also destroyed.

Considerations

The uninstallation mechanism shall counter against downgrade attacks (uninstall the newer application, “fresh install” to an older, potentially more vulnerable version). What is an “older” or “newer” version is defined in ALC_FLR.2.

The secure installation shall destroy all data of the application received after installation of the application. An application developer may mark an application object as exempt from destruction during uninstallation (such as information allowing detection of a downgrade attack). However, decommissioning shall destroy all application data not explicitly marked as exempt of uninstallation. Destruction shall be such that application data cannot be compromised even when the product is also physically accessible to an attacker.

If the application is (partially) defective, it shall still be possible to uninstall the application and then throw the product away, without the application (user) data being recoverable in the product still. Platforms with functionality from “Secure Encrypted Storage”, “Residual Information Purging”, or “Cryptographic KeyStore” typically will be able to fulfil this requirement easier.

The destruction method shall be appropriate for the persistent memory technology and attack potential. See also “Residual Information Purging”.

Traditional CC

[CC Part 2] does not have an equivalent security functional requirement to encode this full feature but FCS_CKM.4 can be used for key destruction or more generally FDP_RIP.1 could be used.

3.2.6 Decommission of Platform

The platform can be decommissioned.

Value

The user invokes this functionality prior to disposing of the product instance or otherwise potentially allowing an attacker physical access to the product instance. As all data is destroyed, the user’s data is also destroyed.

Considerations

The platform shall not be functional after the decommissioning.

The decommissioning shall destroy all data (including the data of the application) received after production. An application developer may mark the application object as exempt from destruction during decommissioning (such that the application code is available even after decommissioning has been performed). However, decommissioning shall destroy all application parts not explicitly marked as exempt of decommissioning. Destruction shall be such that the data and application parts cannot be compromised even when the product is also physically accessible to an attacker.

After destroying all data, the platform may offer a limited diagnostic mode for post-failure analysis.

If the application is (partially) defective, it shall still be possible to decommission the product and then throw the product away, without the (user) data being recoverable in the product still. Platforms with functionality from “Secure Encrypted Storage”, “Residual Information Purging”, or “Cryptographic KeyStore” typically will be able to fulfil this requirement easier.

The destruction method shall be appropriate for the persistent memory technology and attack potential. See also “Residual Information Purging”.

Traditional CC

[CC Part 2] does not have an equivalent security functional requirement to encode this. Commonly this is left as an item to be addressed through manual checks as described in guidance documentation and considered under AGD_OPE.1.

3.2.7 Field Return of Platform

The platform can be returned to the vendor without user data.

Value

The user invokes this functionality prior to returning of the product instance or otherwise potentially allowing an attacker physical access to the product instance. As the user's data is destroyed, the user can be assured that not even the vendor can access personal information.

Considerations

The field return shall destroy all data (including the data of the application) received after user delivery, such that none of this data can be compromised even when the product is also physically accessible to an attacker or the vendor.

This functionality still allows storage of platform instance unique data, such as data needed for "Attestation of Platform Genuineness" and "Attestation of Platform State", allowing the platform and product to be operational still.

This functionality is not intended to counter attacks where an attacker has temporary physical access and then returns it to the user, those are countered by "Physical Attacker Resistance".

Traditional CC

[CC Part 2] does not have an equivalent security functional requirement to encode this. Commonly this is left as an item to be addressed through manual checks as described in guidance documentation and considered under AGD_OPE.1.

3.3 Secure Communication

Typically, a product will use secure channels to communicate with a server, another Connected Product or a cloud service. These SFRs can be used to describe the platform providing such functionality to the application.

Note that the secure communication SFRs should be used when the platform provides the secure channels, the “Cryptographic Operation” SFRs can be used if the platform provides only the cryptographic services that will be used to implement the secure channels.

3.3.1 Secure Communication Support

The platform provides the application with one or more secure communication channel(s).

The secure communication channel authenticates *<list of endpoints>* and protects against *<list of attacks including disclosure, modification, replay, erasure>* of messages between the endpoints, using *<list of protocols and measures>*.

Value

The composite product developer can use the secure communication channels.

The composite evaluator/certifier knows that if the product developer uses the functionality, it is secure in the above manner.

The user has to rely on the composite product developer and composite certifier to know if the secure channels are actually used.

Considerations

The variable parts of this SFR should be completed as follows:

- list of endpoints: The list of endpoints to be protected using the protocols and measures listed hereafter. Endpoints can be identified by their quality, by the interface to which they are connected, or by a more general category, e.g. all local endpoints. If different protocols and measures are used for different endpoints, defining different secure communication channels, then the SFR shall be iterated.
- list of attacks: The list of security issues to avoid, typically including disclosure, modification, replay and erasure.
- list of protocols and measures: The list of the protocols and other measures used to protect the communication channel, e.g. TLS 1.2 with TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8, or IPSEC with …. It is important to provide enough information to properly characterize the quality of the algorithms used.

If multiple different secure communication channels are provided, this SFR should be iterated.

This SFR could be sensitive to misleading claims; the fully authorized SESIP CB and evaluators shall ensure this is not misleading.

Traditional CC

[CC Part 2] has multiple ways to encode communication channels, depending on what type of entity the TOE is communicating with (other portions of the TOE, human, trusted IT entity, etc.). These methods include FPT_ITT Internal TOE TSF data transfer, FTP_ITC Inter-TSF Trust Channel, FTP_TRP Trusted Path, and are sometimes supported by specification of algorithms and key sizes with FCS_COP.

3.3.2 Secure Communication Enforcement

The platform ensures the application can only communicate with *<list of endpoints>* over the secure communication channel(s) supported by the platform using *<list of protocols and measures>*.

Value

The user and composite evaluator/certifier can trust the product is securely communicating (typically with cloud services), if the platform is attested in the secure state.

Considerations

This SFR requires that one or more iterations of the “Secure Communication Support” SFR is claimed.

The first variable part of this SFR, list of endpoints, identifies the secure channels to which the SFR applies, as they have been defined in the corresponding “Secure Communication Support” SFRs. The second variable is to be filled with the name of protocols and measures used for the secure channel implementation,

This SFR implies that the platform does all the unsecured communication layers under the secure communication channel (TCP/IP, DHCP, DNS, BT …), and that the application does not have direct access to the communication layers underlying that secure channel.

This SFR could be sensitive to misleading claims, the fully authorized SESIP CB and evaluators shall ensure this is not misleading.

Traditional CC

[CC Part 2] uses FPT_ITT Internal TOE TSF data transfer to encode communication channels with local endpoints and FTP_ITC Inter-TSF Trust Channel for communication channels with remote endpoints.

3.4 Extra Attacker Resistance

In our regular attacker model, attackers have only logical (network) access to the product, application and platform during the exploitation phase of the attack, and therefore:

- do not have physical access to the specific product instance attacked, and
- are unable to run their own hostile code on the platform.

When this attacker model is not valid, adding security functional requirements from this section allows expression of resistance against physical and software attackers.

Note that during the identification phase, the attacker is assumed to have physical access to his platform instance when preparing his attack.

3.4.1 Limited Physical Attacker Resistance

The platform detects or prevents attacks by an attacker with physical access before the attacker compromises <*list of security functional requirements*>.

Value

For situations where a physical attacker is in scope for a limited set of functionalities such as valuable assets warranting a physical attack, but not for all functionality. The attacker is not limited in his physical attack, the attacker is limited in the target of his attack (the limited set of SFRs).

Considerations

All attacks enabled by physical access within the attack potential need to be considered.

With this SFR, local non-networked interfaces such as an USB port and MicroSD card reader shall now also be considered in the vulnerability analysis as accessible to the attacker (such as an “evil maid”).

Attackers (mis-)using production and debug functionality such as JTAG and ISD functionality would typically be countered by disabling this functionality prior to delivery to the customer. Invasive attacks such as physical tampering or perturbation would typically be countered by detection and decommissioning the product before the detected attack succeeds. Non-invasive attacks such as side channel attacks would typically be countered by not leaking secret information via side channels such as timing, power and EM emissions.

The threat of replacement of the product is not covered by this SFR; it can be countered with SFRs “Verification of Platform Instance Identity” together with “Attestation of Platform Genuineness”, allowing an application or user to detect replacement.

Traditional CC

[CC Part 2] has similar security requirements in the FPT_PHP family, which can be used to specify detection (FPT_PHP.1), notification (FPT_PHP.2) and response (FPT_PHP.3) to physical tampering. Only the traditional FPT_PHP.3 requirement has the capability to limit what attacks to which the TOE should respond.

3.4.2 Physical Attacker Resistance

The platform detects or prevents attacks by an attacker with physical access before the attacker compromises any of the other functional requirements, ensuring that the other functional requirements are not compromised.

Value

For situations where a physical attacker is in scope, such as products that are typically used outside a secured area, or when the products store highly valuable assets warranting a physical attack.

Considerations

All attacks enabled by physical access within the attack potential need to be considered.

With this SFR, local non-networked interfaces such as an USB port and MicroSD card reader shall now also be considered in the vulnerability analysis as accessible to the attacker (such as an “evil maid”).

Attackers (mis-)using production and debug functionality such as JTAG and ISD functionality would typically be countered by disabling this functionality prior to delivery to the customer. Invasive attacks such as physical tampering or perturbation would typically be countered by detection and decommissioning the product before the detected attack succeeds. Non-invasive attacks such as side channel attacks would typically be countered by not leaking secret information via side channels such as timing, power and EM emissions.

The threat of replacement of the product is not covered by this SFR; it can be countered with SFRs “Verification of Platform Instance Identity” together with “Attestation of Platform Genuineness”, allowing an application or user to detect replacement.

Traditional CC

[CC Part 2] has similar security requirements in the FPT_PHP family, which can be used to specify detection (FPT_PHP.1), notification (FPT_PHP.2) and response (FPT_PHP.3) to physical tampering.

3.4.3 Software Attacker Resistance: Isolation of Platform

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 the other functional requirements.

Value

For situations where an attacker may be able to load his own code on the platform, or the attacker might subvert any part of the application.

Considerations

This typically would require at least an OS with kernel-user mode separation.

Traditional CC

[CC Part 2] does not have an equivalent security functional requirement to encode this, although the notion of domain separation is considered under ADV_ARC.1.

3.4.4 Software Attacker Resistance: Isolation of Platform Parts

The platform provides isolation between platform parts, such that an attacker able to run code in *<list of vulnerable platform parts>* can compromise neither the integrity and confidentiality of *<list of protected platform parts>* nor the provision of any other security functional requirements.

Value

For situations where an attacker may be able to load his own code on inner parts of the platform.

The platform developer can separate the critical assets in different parts of the platform, and thus safeguard them from compromises of other parts of the platform.

Considerations

The variable parts of this SFR need to be completed:

- The *list of vulnerable platform parts* defines the parts of the platform that are considered to be potentially compromised by an attacker and used to attack other parts of the platform.
- The *list of protected platform parts* defines the parts of the platform that are considered to be potentially targeted by attackers running code in other compromised parts, and that are protected against such attacks.

A platform offering this functionality would typically also claim “Software Attacker Resistance: Isolation of Platform”.

This would typically require a micro-kernel or a hardware-based isolation technology.

Traditional CC

[CC Part 2] does not have an equivalent security functional requirement to encode this. Although the notion of domain separation is considered under ADV_ARC.1, the isolation between parts of application is not specifically addressed in the traditional CC.

3.4.5 Software Attacker Resistance: Isolation of Application Parts

The platform provides isolation between parts of the application, such that an attacker able to run code as one of the *<list of application parts>* cannot compromise the integrity and confidentiality of the other application parts.

Value

For situations where the product developer wants to separate the critical assets in its own application part (process/executable/…), and thus safeguard them from compromises by other parts of the application code that are too complex to be shown to be secure.

Considerations

The variable part, list of application parts, of this SFR shall list the part types that may compose the application (modules, processes, applets), and between which some isolation is to be provided.

This typically would require an OS with application-application memory separation or an interpreter-based platform with similar access rules. Java Card for example fulfils this.

Traditional CC

[CC Part 2] does not have an equivalent security functional requirement to encode this. Although the notion of domain separation is considered under ADV_ARC.1, the isolation between parts of application is not specifically addressed in the traditional CC.

3.5 Cryptographic Functionality

These are common cryptographic functions that a platform can provide that are useful to a product developer, directly or not visible to the product user.

Standard CC interpretation applies for references to standards: All of the claimed parts of the specification need to be fully implemented, so precise references are encouraged. Evaluators and certifiers shall verify all aspects of the parts of standards referenced.

3.5.1 Cryptographic Operation

The platform provides the application with *<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>*.

Value

Evaluators and developers of composites can be ensured of standard-compliant cryptographic functions.

Considerations

The variable parts in the SFR should be completed as follows:

- The *list of cryptographic operations* defines the operations that can be considered, typically selected among encryption, decryption, hashing, signing, and signature verification.
- The *list of algorithms* provides a reference to the cryptographic algorithms used.
- The *specification* references the standard in which the operation is defined (including a section or similar information if relevant).
- The *list of key lengths* defines the key lengths that are supported for that cryptographic operation.
- The *list of modes* defines the operational modes that are supported for that cryptographic operation.

A typical example of a fully defined SFR would be: The platform provides the application with encryption and decryption functionality as specified in NIST FIPS 197 (AES) with 256-bit keys in GCM mode.

This SFR should be iterated if more than one algorithm is provided. For the sake of clarity, it may be preferable to use a table to define the supported parameters:

Table 3-1: Cryptographic Operations (Example)

Operations	Algorithm	Specification	Key lengths	Modes
Encryption, decryption	AES	NIST FIPS 197	256	GCM

The cryptographic operation shall keep the confidentiality of the secret keys against the attacker. Even with the standard attacker model, timing and padding oracle attacks shall be considered if within the attack potential.

Note that this SFR is for cryptographic functionality available to the application. This SFR should not be used to only restate the crypto functionality claimed in other SFRs (such as “Secure Encrypted Storage”, “Secure Communication Support”, “Secure Communication Enforcement”) unless that functionality is separately made available to the application.

Traditional CC

[CC Part 2] defines the requirement FCS_COP.1 for the specification of cryptographic operations.

3.5.2 Cryptographic Key Generation

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

Value

Evaluators and developers of composites can be ensured of standard-compliant cryptographic key generation.

Considerations

The variable parts in the SFR should be completed as follows:

- The *list of algorithms* provides a reference to the algorithms used
- The *specification* references the standard in which the operation is defined (including a section or similar information if relevant).
- The *list of key lengths* defines the key lengths that are supported for that cryptographic operation.

This SFR should be iterated if keys are generated for more than one algorithm. For the sake of clarity, it may be preferable to use a table to define the supported parameters.

The specification of the key generation algorithm can be useful for use in specific product evaluation schemes. Regardless of the algorithm used, every attack within the attack potential applies: Attacks such as ROCA are always to be checked against.

Traditional CC

[CC Part 2] defines the requirement FCS_CKM.2 for the specification of cryptographic key generation, with FCS_COP.3 for the specification of cryptographic key import.

3.5.3 Cryptographic KeyStore

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

Value

Evaluators and developers of composites can be ensured that keys cannot be disclosed accidentally, provided that the keys are stored only in the KeyStore.

Considerations

The variable parts of the SFR should be completed as follows:

- The *list of assets* defines the types of assets to be protected by the KeyStore.
- The properties *authenticity, integrity, confidentiality* defines the protection afforded.
- The *list of operations* defines the operations that an application can perform on the assets stored in the KeyStore without having to access the assets' values.

A software KeyStore in the platform typically will require either "Software Attacker Resistance: Isolation of Platform" in the platform or code review or automated code verification of the product.

Traditional CC

[CC Part 2] does not have a singular way in which the secure key storage is specified. Typically, it is defined through a combination of user data protection (access control policy FDP_ACC.1 and access control functions FDP_ACF.1) together with requirement FCS_COP.1 for cryptographic operations.

3.5.4 Cryptographic Random Number Generation

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

Value

Evaluators and developers of composites can be ensured of standard-compliant cryptographic key generation.

Considerations

The variable parts of the SFR should be completed as follows:

- The list of entropy sources defines how the randomness is generated from physical and computational resources.
- The specification references the standard in which the operation is defined (including a section or similar information if relevant).

The specification of the number generation algorithm can be useful for use in specific product evaluation schemes. Regardless of the algorithm used, every attack within the attack potential applies: Weak entropy and predictability shall always be checked against.

Traditional CC

Traditional CC does not have an equivalent security requirement. [PP-0084] has the SFR FCS_RNG.1 with open operations allowing encoding as such:

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

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

3.6 Compliance Functionality

These are commonly required properties from various product domains and schemes.

3.6.1 Secure Storage

The platform ensures that all data stored by the application, except for <*list of data stored in plaintext*>, is protected to ensure its authenticity and integrity as specified in <*specification*> with a platform instance unique key of key length <*key length*>.

Value

Evaluators and developers of composites can be ensured that stored data is integer and authentic.

Considerations

The variable parts of the SFR should be completed as follows:

- The list of data stored as plaintext lists the categories that are excluded from the secure encrypted storage, which should be used for all other data.
- The specification references the standard in which the cryptographic algorithms used are defined (including a section or similar information if relevant).
- The key length is the size of the keys used by the cryptographic algorithms used.

The mechanism used shall protect both the authenticity and the integrity of stored data.

The key used in this mechanism shall be generated such that compromise of the key of one product-instance does not allow easier compromise of another product-instance.

This may be implemented using the “Cryptographic Operation” functionality, but that SFR should only be claimed if the functionality of that SFR is directly available to the application.

Note that if the storage must be protected against read out by physical attackers, “Physical Attacker Resistance” should be used.

Traditional CC

[CC Part 2] defines FDP_SDI.1/2 for the integrity protection and FDP_DAU.1 for authenticity protection of user data; it also, defines FCS_COP.1 for cryptographic operations involved.

3.6.2 Secure Encrypted Storage

The platform ensures that all data stored by the application, 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*>.

Value

Evaluators and developers of composites can be ensured that (implicitly) stored data is encrypted, without further activities.

Considerations

The variable parts of the SFR should be completed as follows:

- The list of data stored as plaintext lists the categories that are excluded from the secure encrypted storage, which should be used for all other data.
- The specification references the standard in which the encryption mechanism is defined (including a section or similar information if relevant).
- The key length is the size of the key used for the encryption.

The encryption mechanism used shall protect both the integrity and confidentiality of stored data.

The key used to encrypt the data shall be generated such that compromise of the key of one product-instance does not allow easier compromise of another product-instance; moreover, the confidentiality of this key must be protected.

This may be implemented using the “Cryptographic Operation” functionality, but that SFR should only be claimed if the functionality of that SFR is directly available to the application.

Traditional CC

[CC Part 2] does not have a singular way in which the secure encrypted storage is specified. Typically, it is defined through a combination of user data protection (access control policy FDP_ACC.1 and access control functions FDP_ACF.1) together with requirement FCS_COP.1 for cryptographic operations.

3.6.3 Secure External Storage

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 <*authenticity, integrity, confidentiality, binding to the platform instance, versioning*> is ensured.

Value

Evaluators and developers of composites can be ensured that data can be stored outside of the platform without being affected, neither modified nor disclosed.

Considerations

The variable parts of the SFR should be completed as follows:

- The list of data to be stored outside of the direct control of the platform.
- The protections ensured by the mechanism.

If cryptographic algorithms are used, also provides:

- The specification references the standard in which the cryptographic algorithms used for data encryption are defined (including a section or similar information if relevant).
- The key length is the size of the keys used by the cryptographic algorithms used for data encryption.

The key used in the encryption mechanism shall be generated such that compromise of the key of one product-instance does not allow easier compromise of another product-instance.

This may be implemented using the “Cryptographic Operation” functionality, but that SFR should only be claimed if the functionality of that SFR is directly available to the application.

Note that if the storage must be protected against read out by physical attackers, “Physical Attacker Resistance” should be used.

Traditional CC

[CC Part 2] defines FDP_SDI.1/2 for the integrity protection and FDP_DAU.1 for authenticity protection of user data; it also, defines FCS_COP.1 for cryptographic operations involved. No SFR is directly defined to cover binding and versioning concerns.

3.6.4 Residual Information Purging

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 (re)used by the platform or application again and before an attacker can access it.

Value

Evaluators and developers of composites can be ensured that (implicitly) stored and copied data is erased automatically, without further activities.

Considerations

The variable parts of the SFR should be completed as follows:

- The *list of data* defines the categories of data that are to be cleared automatically, which is expected to be very wide (see below).
- The *list of exceptions of data that is not erased automatically* defines the categories that are not expected to be cleared automatically.
- The *specification* references the standard in which the encryption mechanism is defined (including a section or similar information if relevant)

Typically, list of data would be “all data of the application no longer used by the application”, allowing for lazy erasure happening only at re-use of the memory, not immediately at release of the memory to match common software implementations.

Attacks such as cold boot need to be considered.

Traditional CC

[CC Part 2] defines a similar SFR in FDP_RIP for residual information protection.

3.6.5 Audit Log Generation and Storage

The platform generates and maintains an audit log of <*list of significant security events*> and allows access and analysis of these logs following a specific <*access control policy*>.

Value

Evaluators, developers and users of composites can detect attack attempts to the platform.

Considerations

The variable parts of the SFR should be completed as follows:

- The *list of significant security events* defines the events in the log, which are expected to provide a wide coverage of the platform’s possible events.
- The *access control policy* defines the conditions under which the logs may be inspected, typically by specific privileged users after authentication.

“Software Attacker Resistance: Isolation of Platform” might also be claimed to support this claim.

Traditional CC

[CC Part 2] defines a similar SFR of FAU_GEN.1 for generation of audit logs.

3.6.6 Reliable Index

The platform implements a strictly increasing function.

Value

This feature can be used to implement means to track versions or even time.

Considerations

An only-increasing counter of events or timer ticks fulfills this requirement, as does a protected time measurement. The function must not allow reset of the value, even in the context of a “Factory Reset of Platform”.

Traditional CC

[CC Part 2] defines a similar SFR of FPT_STM.1 for provision of reliable time stamp for use by the TOE.

3.6.7 Secure Debugging

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

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

Value

Developers of composites can debug their applications without compromising the security of their users’ data.

Considerations

The variable parts of the SFR can be completed as follows:

- The *list of endpoints* lists the endpoints (potentially with their physical connection and the authentication data needed) allowed to set the platform in debug mode.
- The *specification* references the standard in which the authentication mechanism is defined (including a section or similar information if relevant).
- The *list of exceptions* lists the data that is excluded from the protection during debugging. This should not include application data that has a reasonable expectation of containing the user’s personally identifiable information.

This may be implemented using “Secure Communication Support” functionality.

Traditional CC

[CC Part 2] does not have a singular way in which the secure debug is specified. Typically, it is partly defined through a combination of FPT_ITT Internal TOE TSF data transfer, FTP_ITC Inter-TSF Trust Channel, FTP_TRP Trusted Path, and are sometimes supported by specification of algorithms and key sizes with FCS_COP.

4 Security Assurance Requirements

This document contains five hierarchical sets of Common Criteria assurance packages that are suitable to evaluate IoT platforms or parts thereof.

The sets are named **SESIP1**, **SESIP2**, **SESIP3**, **SESIP4** and **SESIP5** and are hierarchical:

- **SESIP Assurance Level 1 (SESIP1)** is a self-assessment-based level: The developer shall provide a simplified Security Target, describing the security claims of his product, together with a compliance rationale why he believes these claims are met. Only minimal evaluator effort is needed: Checks on consistency and clarity of these compliance rationales are performed. There is no independent check by the evaluators the platform implements the SFRs. SESIP1 provides a basic level of assurance.
- **SESIP Assurance Level 2 (SESIP2)** is a black-box penetration testing level: The evaluation is structured around a time-limited penetration testing effort. No design or source code is required to be available besides a full functional specification. This is the highest level that can be applied to a closed-source platform without cooperation by the developer. SESIP2 provides a moderate level of assurance.
- **SESIP Assurance Level 3 (SESIP3)** is a traditional white-box vulnerability analysis: The evaluation is structured around a time-limited source code analysis combined with a time-limited penetration testing effort. Other assurance components have only been included to support this approach to save as much effort as possible. SESIP3 provides a substantial level of assurance.
- **SESIP Assurance Level 4 (SESIP4)** is exclusively for re-use of SOG-IS certified platforms or platform parts by licensed evaluation laboratories, allowing those platforms to utilize the mappings from SESIP to specific commercial product domains. A SESIP4 evaluation must then be performed as a complement to a SOG-IS certification that includes at least all the standard Common Criteria assurance components, and in particular AVA_VAN.4. The current methodology simply provides guidance on how to obtain a SESIP4 certificate in addition to such a SOG-IS certificate.
- **SESIP Assurance Level 5 (SESIP5)** is exclusively for re-use of SOG-IS certified platforms or platform parts by licensed evaluation laboratories, allowing those platforms to utilize the mappings from SESIP to specific commercial product domains. A SESIP5 evaluation must then be performed as a complement to a SOG-IS certification that includes at least all the standard Common Criteria assurance components, and in particular AVA_VAN.5 The current methodology simply provides guidance on how to obtain a SESIP5 certificate in addition to such a SOG-IS certificate.

Augmentation of the assurance requirements is not allowed in SESIP, so a platform developer cannot claim compliance to a component of higher assurance than those specified in the SESIP level mentioned in the Security Target.

However, the SESIP-accredited certification bodies of a certification scheme may decide to add a SAR or use a higher-level SAR to any SESIP level, provided that these criteria apply to all certifications performed under that scheme.

Note:

In the description of the levels, most Security Assurance Requirements (SARs) are from Common Criteria. The SESIP-specific SARs are indicated using a **bold** font, and the SESIP-refined SARs are indicated using an *italic* font.

4.1 SESIP Assurance Level 1 (SESIP1)

SESIP1 provides a basic level of assurance for IoT platforms. SESIP1 is based on self-assessment by the developer, with only minimal verification by an evaluator. There is no independent check by the evaluators that the platform implements the SFRs.

At this assurance level, the developer is expected to provide:

- A simplified Security Target with compliance rationale based on the developer's self-assessment (ST templates are included as annex). The compliance rationale includes:
  - a reference to (self-)check for publicly known vulnerabilities against the platform,
  - references to guidance documents describing the objectives for the environment,
  - a reference to the public facing procedures describing how flaws are reported, tracked and corrected, and the resulting updates communicated,
  - a description of the coverage of every SFR by the platform's security features and the way in which these features' security characteristics have been assessed by the developer, and
  - in the case of a composite platform, a description of the way the environment objectives are met for every platform part used in the composite platform.
- The referred (self-)check against publicly known vulnerabilities and the referred guidance documents. The public facing procedures need to be available to the public (and hence the evaluators) already.
- Internal Flaw Reporting procedures describing internal (as well as the public facing) procedures describing how flaws are reported, tracked and corrected, and the resulting updates communicated.

4.1.1 Objectives

SESIP1 provides assurance by using a simplified Security Target, augmented by a self-assessment rationale. The self-assessment is part of the Security Target, even if it is delivered separately (for example in the form of a completed questionnaire). The self-assessment includes testing of the TOE based on a survey of public domain sources for potential vulnerabilities.

The evaluator assesses the Security Target for clarity and consistency, verifies that the referenced security features are described in the documentation, but the existence or effectiveness of the security functionality in the actual platform is not independently assessed by the evaluator or the certifier.

4.1.2 Assurance Components

Table 4-1: SESIP1 Assurance Requirements

Assurance Class	Assurance Families
ASE: Security Target evaluation	ASE_INT.1 – ST Introduction *ASE_OBJ.1 – Security requirements for the operational environment* **ASE_REQ.3 – Listed security requirements** *ASE_TSS.1 – TOE summary specification*
AGD: Guidance documents	AGD_OPE.1 – Operational user guidance AGD_PRE.1 – Preparative procedures
ALC: Life-cycle support	ALC_FLR.2 – Flaw reporting procedures
AVA: Vulnerability Assessment	AVA_VAN.1 – Vulnerability survey

4.1.3 Security Target Requirements

ASE_INT.1 ST Introduction

As per [CC Part 3].

Dependencies:

- No dependencies.

Refinement:

The ST should also list the SESIP version.

ASE_OBJ.1 Security objectives for the operational environment

Dependencies:

- No dependencies.

Developer action elements:

ASE_OBJ.1.1D the developer shall provide a statement of security objectives.

Content and presentation elements:

ASE_OBJ.1.1C the statement of security objectives shall describe the security objectives for the operational environment.

Refinement:

All security objectives concerning the operational environment shall be listed with references to the place in the guidance documents where these objectives are addressed.

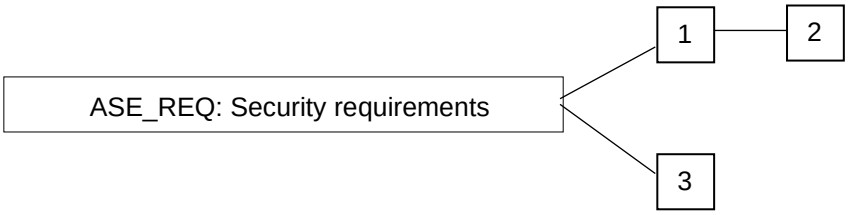
Evaluator action elements:

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

ASE_REQ.3 Listed security requirements

Amended hierarchy of components in ASE_REQ family:

Figure 4-1: ASE_REQ Component Hierarchy



Dependencies:

- No dependencies.

Developer action elements:

ASE_REQ.3.1D the developer shall provide a statement of security requirements.

Content and presentation elements:

ASE_REQ.3.1C the statement of security requirements shall describe the SFRs and the SARs.

ASE_REQ.3.2C All SFRs shall be drawn from the list of allowed Security Functional Requirements.

ASE_REQ.3.3C The SFR "Verification of Platform Identity" shall be included. The SFR "Secure Update of Platform" shall be included or under the ALC_FLR.2 it shall be argued why updates are not applicable.

ASE_REQ.3.4C The SARs shall be an exact SESIP assurance level. No augmentation is allowed.

ASE_REQ.3.5C If multiple SESIP assurance levels are claimed, it shall be clear to the reader of the ST what SFRs covered by what SESIP assurance level.

Evaluator action elements:

ASE_REQ.3.1E the evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

Guidance:

If multiple instances of a claim for security functionality are required, the body of an SFRs may be iterated. For example, if multiple secure communication channels are supported the "Secure Communication Support" may be iterated as follows:

The secure communication channel:

1. authenticates **servers** and protects against **disclosure, modification** of messages between the endpoints, using **TLS 1.2 with TLS_RSA_WITH_AES_128_CBC_SHA cipher suite**
2. authenticates **servers and clients** and protects against **disclosure, modification** of messages between the endpoints, using **SSH 1.2 with aes256-cbc using hmac-sha2-256 for SSH transport, rsa-sha2-256 SSH public-key based authentication and diffie-hellman-group14-sha256 key exchange**.

If an SFR makes references to an external standard, the reference shall be precise. By including a reference to a standard, everything that could reasonably be considered part of the reference implicitly forms part of the SFR claim and shall therefore be verified. For example, a reference to FIPS-140-2 requires FIPS-140-2 certification. Whereas a reference to FIPS-140-2 section 4.9.1 does not require FIPS certification but does require power-up tests to be performed by the TOE, including cryptographic algorithm tests, software/firmware integrity test and critical functions tests.

The evaluator determines that the statement of security assurance requirements matches one of the SESIP assurance levels defined in Chapter 4 of this document. If the SARs are listed or copied in the ST, the evaluator determines that the set of claimed SARs exactly matches one of the SESIP assurance levels defined in SESIP (this document). It is not permitted to extend or augment a SESIP assurance level.

The evaluator determines that it is clear to the reader of the ST what SFRs are covered by what SARs. Multiple claims of SESIP assurance levels are allowed (as described in "Additive Composition within SESIP"), provided the claims are clear.

ASE_TSS.1 TOE summary specification

As per [CC Part 3].

Dependencies:

The dependency on ASE_REQ.1 is fulfilled by ASE_REQ.3.

Refinement:

The TSS shall include a self-assessment by the developer how the TOE correctly implements the Security Functional Requirements.

This self-assessment may be provided as a separate document (such as a filled-in questionnaire), however is considered part of the public ST.

The self-assessment shall consider each implementation of the Security Functional Requirements, describing how the developer has supported his assessment, based, for instance on:

- *Testing, either by the developer or by a third party.*
- *Conformance to another standard by the TOE or part of the TOE.*
- *Reliance on the statements of another party on the TOE or a part of the TOE.*

The self-assessment shall indicate what procedures cover ALC_FLR.2 if "Secure Update of Platform" is included in the Security Target.

4.1.4 Guidance Documents Requirements

AGD_OPE.1 Operational User Guidance

As per [CC Part 3].

Dependencies:

- ADV_FSP.1 is met by ASE_REQ.3 and the refinement of ASE_TSS.1.

AGD_PRE.1 Preparative Procedures

As per [CC Part 3].

Dependencies:

- ADV_FSP.1 is met by ASE_REQ.3 and the refinement of ASE_TSS.1.

4.1.5 Life-cycle Support Requirements

ALC_FLR.2 Flaw Reporting Procedures

As per [CC Part 3].

Dependencies:

- No dependencies

Refinement:

It is recognized that many IoT platforms will require a method to update in the field to defend against new threats or against vulnerabilities that were found later on. On the other hand, there may also be IoT platforms for which this would be impractical or overkill (e.g. a very simple low-power sensor that communicates only one way).

To enable both TOE types, this assurance requirement can be met in two ways:

- *For platforms that can be updated, as per [CC Part 3]. Note that these platforms shall have the SFR "Secure Update of Platform" included in the Security Target.*
- *For platforms that cannot be updated, the Security Target shall contain a rationale why it is acceptable that this platform cannot be updated in the field. It is acceptable that a hardware root of trust cannot be updated in the field.*

In both cases, the flaw reporting procedure shall describe how flaws are to be reported to the developer, and how updates to the platform or guidance (or retraction of the product as certified) are communicated to the users of the platform.

At SESIP1 and SESIP2 level, flaw reporting procedure steps not visible to the users may be omitted from the description as these are not verifiable by the evaluators. The public facing procedures shall be described.

4.1.6 Vulnerability Assessment Requirements

AVA_VAN.1 Vulnerability Survey

As per [CC Part 3].

Dependencies:

- AGD_OPE.1 and AGD_PRE.1 are met by ASE_TSS.1 and the availability of the guidance documents.
- ADV_FSP.1 is met by ASE_TSS.1.

Refinement:

The vulnerability survey may be performed by the developer or evaluator.

The evaluator is to determine whether the testing evidence submitted by the developer considers all applicable public domain sources to identify potential vulnerabilities in the TOE. This testing may utilize appropriate penetration test tools available in the public domain to test for publicized potential vulnerabilities.

4.2 SESIP Assurance Level 2 (SESIP2)

SESIP2 is a moderate level of assurance for (parts of) IoT platforms.

SESIP2 provides significantly more assurance than SESIP1 by requiring a vulnerability analysis and actual penetration testing on the platform by an evaluator but will provide less assurance than the higher assurance provided by white-box SESIP3.

At this assurance level, the developer is expected to provide:

- A simplified Security Target with compliance rationale based on the developer's self-assessment (ST templates are included as annex). The compliance rationale includes:
  - references to guidance documents describing the objectives for the environment,
  - a reference to the public facing procedures describing how flaws are reported, tracked and corrected, and the resulting updates communicated,
  - a description of the coverage of every SFR by the platform's security features and the way in which these features' security characteristics have been assessed by the developer, and
  - in the case of a composite platform, a description of the way the environment objectives are met for every platform part used in the composite platform.
- Guidance documents. This typically consists of the existing user manuals and data sheets.
- A complete functional specification. This typically consists of existing programmer's manuals, data books or API specifications, and a mapping showing what interfaces implement the SFRs declared in the Security Target.
- Proof of functional conformance testing.
- Internal Flaw Reporting procedures describing internal (as well as the public facing) procedures describing how flaws are reported, tracked and corrected, and the resulting updates communicated.

4.2.1 Objectives

SESIP2 provides assurance by an analysis of the SFRs in the simplified Security Target, using a full functional specification, guidance documentation, and the platform being tested, to understand the security behavior.

The analysis is supported by independent testing of the platform, and a vulnerability analysis demonstrating resistance to penetration attackers with a Basic attack potential. The vulnerability analysis is based upon mapping of SFRs to interfaces and guidance evidence provided, and the description of all interfaces as provided by the functional specification.

SESIP2 also provides assurance through the assessment of the developer's procedures to produce and distribute updates to IoT Platforms in the field (ALC_FLR.2).

4.2.2 Assurance Components

Table 4-2: SESIP2 Assurance Requirements

Assurance Class	Assurance Families
ASE: Security Target evaluation	ASE_INT.1 – ST Introduction *ASE_OBJ.1 – Security requirements for the operational environment* **ASE_REQ.3 – Listed security requirements** *ASE_TSS.1 – TOE summary specification*
ADV: Development	ADV_FSP_4 – Complete functional specification
AGD: Guidance documents	AGD_OPE.1 – Operational user guidance AGD_PRE.1 – Preparative procedures
ALC: Life-cycle support	ALC_FLR.2 – Flaw reporting procedures
ATE: Tests	ATE_IND.1 – Independent testing: conformance
AVA: Vulnerability Assessment	AVA_VAN.2 – Vulnerability analysis

4.2.3 Security Target Requirements

As per section 4.1.3.

4.2.4 Development Requirements

ADV_FSP.4 Complete functional specification

As per [CC Part 3].

Dependencies:

- ADV_TDS.1 Basic design is considered to be sufficiently fulfilled by ASE_TSS.1 for the black-box platform.

Note:

The objective of ADV_FSP.4 in the SESIP context is to ensure that the platform developer has precisely specified all the platform's interfaces with a sufficient level of details, including direct error messages, as described in the Developer action elements and in the Content and presentation elements in the definition of ADV_FSP.4 in [CC Part 3].

The evaluator, as also indicate in [CC Part 3], shall then verify that the provided specification contains the required elements, and that it provides a full coverage of the SFRs, with the help of the mapping provided by the developer.

Since ADV_TDS.1 is not required, there is no link to be considered with the high-level design.

4.2.5 Guidance Documents Requirements

As per section 4.1.4.

4.2.6 Life-cycle Support Requirements

As per section 4.1.5.

4.2.7 Tests Requirements

ATE_IND.1 Independent testing: conformance

As per [CC Part 3].

Note:

Re-use of industry standard testing by the developer or third parties is encouraged, provided this is accepted by the CB.

4.2.8 Vulnerability Analysis Requirements

AVA_VAN.2 Vulnerability analysis

As per [CC Part 3].

Dependencies:

- AGD_OPE.1 and AGD_PRE.1 are met.
- ADV_ARC.1 and ADV_TDS.1 are met by ASE_TSS.1 in combination with ADV_FSP.4.
- ADV_FSP.2 is met by ADV_FSP.4.

Note:

ADV_IMP package not being claimed at this SESIP level does not forbid the vulnerability assessment to be based on source code review if provided; this may reduce the testing effort. In any case, the use of source code review in vulnerability analysis should be clearly mentioned in the evaluation report.

In case of a composite platform, the evaluator shall verify that all objectives for the environment of the platform parts fulfilled by another platform part are accurately fulfilled. All guidance of one platform part for another platform part should be considered as part of the vulnerability analysis.

4.3 SESIP Assurance Level 3 (SESIP3)

SESIP3 is a substantial level of assurance for (parts of) IoT platforms.

It provides significantly more assurance than SESIP2 by requiring significant source code analysis by an evaluator as input to the vulnerability analysis. Use of the source code analysis will increase the assurance gained from the vulnerability analysis and penetration testing but will provide less assurance than the extended-substantial assurance provided by SESIP4.

At this assurance level, the developer is expected to provide:

- A simplified Security Target with compliance rationale based on the developer's self-assessment (ST templates are included as annex). The compliance rationale includes:
  - references to guidance documents describing the objectives for the environment,
  - a reference to the public facing procedures describing how flaws are reported, tracked and corrected, and the resulting updates communicated,
  - a description of the coverage of every SFR by the platform's security features and the way in which these features' security characteristics have been assessed by the developer, and
  - in the case of a composite platform, a description of the way the environment objectives are met for every platform part used in the composite platform.
- Guidance documents. This typically consists of the existing user manuals and data sheets.
- A complete functional specification. This typically consists of existing programmer's manuals, data books or API specifications, and a mapping showing what interfaces implement the SFRs declared in the Security Target.
- The full source code, and a mapping showing where in the source code the SFRs are implemented. This mapping typically is a minor extension of the mapping for the functional specification.
- Proof of functional conformance testing.
- A short description showing how the platform's version number is maintained to be uniquely identifying the platform in its version and showing that the whole platform (source code and all that is described above) is maintained in a standard version management system (such as CVS, GIT or SVN).
- Internal Flaw Reporting procedures describing internal (as well as the public facing) procedures describing how flaws are reported, tracked and corrected, and the resulting updates communicated.

4.3.1 Objectives

SESIP3 provides assurance by a simplified security target, and an analysis of the SFRs in that ST, using a full functional specification, guidance documentation, and the implementation representation, to understand the security behavior.

The analysis is supported by independent testing of the TSF, and a vulnerability analysis (based upon the functional specification, implementation representation and guidance evidence provided) demonstrating resistance to penetration attackers with an Enhanced-Basic attack potential.

SESIP3 also provides assurance using development environment controls and additional TOE configuration management including automation, and evidence of secure delivery procedures.

4.3.2 Assurance Components

Table 4-3: SESIP3 Assurance Requirements

Assurance Class	Assurance Families
ASE: Security Target evaluation	ASE_INT.1 – ST Introduction *ASE_OBJ.1 – Security requirements for the operational environment* **ASE_REQ.3 – Listed security requirements** *ASE_TSS.1 – TOE summary specification*
ADV: Development	ADV_FSP_4 – Complete functional specification **ADV_IMP.3 – Complete mapping of the implementation representation of the TSF to the SFRs**
AGD: Guidance documents	AGD_OPE.1 – Operational user guidance AGD_PRE.1 – Preparative procedures
ALC: Life-cycle support	ALC_CMC.1 – Labelling of the TOE ALC_CMS.1 – TOE CM coverage ALC_FLR.2 – Flaw reporting procedures
ATE: Tests	ATE_IND.1 – Independent testing: conformance
AVA: Vulnerability Assessment	AVA_VAN.3 – Focused vulnerability analysis

4.3.3 Security Target Requirements

As per section 3.1.3.

4.3.4 Development Requirements

ADV_FSP.4 Complete functional specification

As per [CC Part 3].

Dependencies:

- ADV_TDS.1 Basic design does not need to be fulfilled, as ADV_IMP.3.3D covers this.

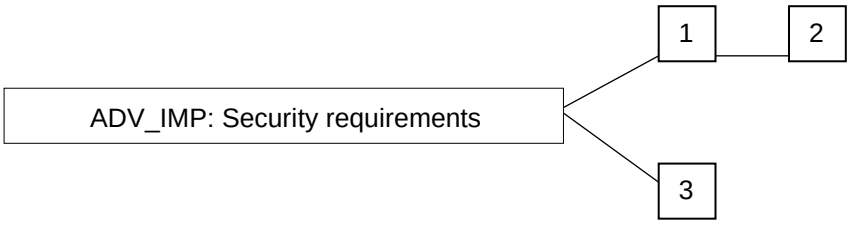
ADV_IMP.3 Complete mapping of the implementation representation of the TSF to the SFRs

Dependencies:

- ASE_REQ.3 Listed security requirements

Amended hierarchy of components in ADV_IMP family:

Figure 4-2: ADV_IMP Component Hierarchy



Developer action elements:

ADV_IMP.3.1D the developer shall make available the implementation representation for the entire TSF.

ADV_IMP.3.2D the developer shall provide a mapping between the SFRs and the entire implementation representation.

ADV_IMP.3.3D the developer shall provide further information on the structure and meaning of the implementation representation, when so required by the evaluator.

Content and presentation elements:

ADV_IMP.3.1C the implementation representation shall define the TSF to a level of detail such that the TSF can be generated without further design decisions.

ADV_IMP.3.2C the implementation representation shall be in the form used by the development personnel.

Evaluator action elements:

ADV_IMP.3.1E the evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_IMP.3.2E the evaluator shall determine that the implementation representation is an accurate and complete instantiation of the SFRs.

See also the refinement for “Vulnerability Analysis Requirements”.

4.3.5 Guidance Documents Requirements

As per section 4.1.4.

4.3.6 Life-cycle Support Requirements

ALC_CMC.1 Labelling of the TOE

As per [CC Part 3].

ALC_CMS.1 TOE CM Coverage

As per [CC Part 3].

ALC_FLR.2 Flaw Reporting Procedures

As per section 3.1.4.

Note:

At SESIP3 level and higher, the entire flaw reporting procedure is to be described; not just the externally visible steps required for SESIP1 and SESIP2 levels.

4.3.7 Tests Requirements

As per section 4.2.7.

4.3.8 Vulnerability Analysis Requirements

AVA_VAN.3 Focused vulnerability analysis

As per [CC Part 3].

Dependencies:

- AGD_OPE.1 and AGD_PRE.1 are met.
- ADV_FSP.2 is met by ADV_FSP.4.
- ADV_ARC.1 and ADV_TDS.3 are met by ADV_IMP.3 in combination with ASE_TSS.1.
- ADV_IMP.1 is met by ADV_IMP.3
- ATE_DPT.1 is met through a combination of ATE_IND.1 and ADV_IMP.3 informing the selection of independent test cases to ensure all major parts (subsystems) of the TOE have been tested.

Notes:

In case of a composite platform, the evaluator shall verify that all objectives for the environment of the platform parts fulfilled by another platform part are accurately fulfilled. All guidance of one platform part for another platform part should be considered as part of the vulnerability analysis.

4.4 SESIP Assurance Level 4 (SESIP4)

SESIP4 is an extended-substantial level of assurance for (parts of) IoT platforms. It is intended to provide significantly more assurance than SESIP3 by raising the required product resistance level, adding design implementation analysis, and strengthening the production environment assessment.

A SESIP4 evaluation must be performed as a complement to a SOG-IS certification that includes at least all the standard Common Criteria assurance components listed below, and in particular AVA_VAN.4. The current methodology simply provides guidance on how to obtain a SESIP4 certificate in addition to such a SOG-IS certificate.

4.4.1 Objectives

SESIP4 provides assurance by a simplified security target, and an analysis of the SFRs in that ST, using a full functional specification, guidance documentation, and the implementation representation, to understand the security behavior.

The analysis is supported by independent testing of the TSF, selective independent confirmation of the developer test results and a vulnerability analysis (based upon the functional specification, implementation representation and guidance evidence provided) demonstrating resistance to penetration attackers with a moderate attack potential.

SESIP4 also provides assurance using development environment controls and additional TOE configuration management including automation and definition of tools used, evidence of secure delivery procedures and site security procedures.

In terms of evaluation, in this version of the standard, a SESIP4 is intended to complement a previous Common Criteria evaluation performed by a SOG-IS accredited laboratory. The purpose of the SESIP4 evaluation therefore is to import the Common Criteria certificate into the SESIP4 ecosystem, which is obtained by writing a SESIP Security Target.

In a SESIP4 evaluation, the evaluator does not repeat the work of the CC evaluation; instead, the evaluator's focus is on the verification of the translation of the CC Security Target into the SESIP Security Target. This is expected to be a straightforward task: In most cases, the CC and SESIP evaluators should be the same entity, and the CC Security Target should be structured in a way that eases its translation into SESIP, for instance by using wording close to the SESIP SFRs in the *TOE Summary Specification* section.

In addition to the CC Security Target content (see next section), a SESIP4 Security Target must also include the rationale required for every SESIP evaluation:

- references to guidance documents describing the objectives for the environment,
- a reference to the public facing procedures describing how flaws are reported, tracked and corrected, and the resulting updates communicated,
- in the case of a composite platform, a description of the way the environment objectives are met for every platform part used in the composite platform.

The SESIP-specific elements of the Security Target may be included in a dedicated document and evaluated separately, preferably by the same laboratory, under the responsibility of a SESIP scheme's certification authority.

4.4.2 Assurance Components

Table 4-4: SESIP4 Assurance Requirements

Assurance Class	Assurance Families
ASE: Security Target evaluation	ASE_INT.1 – ST Introduction *ASE_OBJ.1 – Security requirements for the operational environment* **ASE_REQ.3 – Listed security requirements** *ASE_TSS.1 – TOE summary specification*
ADV: Development	ADV_ARC.1 Security architecture description ADV_FSP_4 – Complete functional specification **ADV_IMP.3 – Complete mapping of the implementation representation of the TSF to the SFRs**
AGD: Guidance documents	AGD_OPE.1 – Operational user guidance AGD_PRE.1 – Preparative procedures
ALC: Life-cycle support	ALC_CMC.1 – Labelling of the TOE ALC_CMS.1 – TOE CM coverage ALC_DEL.1 Delivery procedures ALC_DVS.1 Identification of security measures ALC_FLR.2 – Flaw reporting procedures ALC_TAT.1 Well-defined development tools
ATE: Tests	ATE_COV.1 Evidence of coverage ATE_FUN.1 Functional testing ATE_IND.1 – Independent testing: conformance
AVA: Vulnerability Assessment	AVA_VAN.4 – Methodical vulnerability analysis

4.4.3 Security Target Requirements

As per section 4.3.3.

Notes:

The Security Target introduction (ASE_TSS.1) must refer to the prerequisite Common Criteria certification, in particular by including the certification body and certificate number.

In order to achieve the accessibility and reusability objectives, SESIP SFRs must be listed in the Security Target, and the associated compliance rationale must include a reference to the SFRs included in the original CC Security Target.

The TOE summary specification must be expressed as in other SESIP levels, typically “embedded” in the list of SFRs. Developers can reuse the content provided in the original CC Security Target, provided that they ensure that this content meets SESIP’s accessibility requirements.

ASE_OBJ.1 Security objectives for the operational environment

As per section 4.3.3.

ASE_REQ.3 Listed security requirements

As per section 4.3.3.

Note:

In order to achieve the accessibility and reusability objectives, SESIP SFRs must be listed in the Security Target, and the associated compliance rationale must include a reference to the SFRs included in the original CC Security Target.

ASE_TSS.1 TOE summary specification

As per section 4.3.3.

Note:

The TOE summary specification must be expressed as in other SESIP levels, typically “embedded” in the list of SFRs. Developers can reuse the content provided in the original CC Security Target, provided that they ensure that this content meets SESIP’s accessibility requirements.

4.4.4 Development Requirements

As per [CC Part 3] for all ADV requirements listed in Table 4-4.

4.4.5 Guidance Documents Requirements

As per [CC Part 3] for all AGD requirements listed in Table 4-4.

4.4.6 Life-cycle Support Requirements

As per [CC Part 3] for all ALC requirements listed in Table 4-4.

4.4.7 Tests Requirements

As per [CC Part 3] for all ATE requirements listed in Table 4-4.

4.4.8 Vulnerability Analysis Requirements

As per [CC Part 3] for all AVA requirements listed in Table 4-4.

4.5 SESIP Assurance Level 5 (SESIP5)

SESIP5 is the same as the standard high assurance level currently being used for smartcards, secure elements, e-passports, etc. It provides a very robust defense against very advanced threats.

A SESIP5 evaluation must be performed as a complement to a SOG-IS certification that includes at least all the standard Common Criteria assurance components listed below, and in particular AVA_VAN.5. The current methodology simply provides guidance on how to obtain a SESIP4 certificate in addition to such a SOG-IS certificate.

4.5.1 Objectives

SESIP5 provides assurance by a full security target and an analysis of the SFRs in that ST, using a functional and complete interface specification, guidance documentation, a description of the basic modular design of the TOE, and the entire implementation, to understand the security behavior.

The analysis is supported by independent testing of the TSF, evidence of developer testing based on the functional specification and TOE design, selective independent confirmation of the developer test results, and a vulnerability analysis (based upon the functional specification, TOE design, implementation representation, security architecture description and guidance evidence provided) demonstrating resistance to penetration attackers with a High attack potential.

SESIP5 also provides assurance using development environment controls and additional TOE configuration management including automation, evidence of secure delivery procedures, and, where possible procedures for updating the TOE in the field.

SESIP5 includes the SESIP-specific ASE_REQ.3 assurance requirement, which establishes a link between the SOG-IS certification and the SESIP ecosystem, by mapping the Common Criteria SFRs to SESIP SFRs.

In terms of evaluation, in this version of the standard, a SESIP5 is intended to complement a previous Common Criteria evaluation performed by a SOG-IS accredited laboratory. The purpose of the SESIP5 evaluation therefore is to import the Common Criteria certificate into the SESIP5 ecosystem, which is obtained by writing a SESIP Security Target.

In a SESIP5 evaluation, the evaluator does not repeat the work of the CC evaluation; instead, the evaluator's focus is on the verification of the translation of the CC Security Target into the SESIP Security Target. This is expected to be a straightforward task: In most cases, the CC and SESIP evaluators should be the same entity, and the CC Security Target should be structured in a way that eases its translation into SESIP, for instance by using wording close to the SESIP SFRs in the *TOE Summary Specification* section.

In addition to the CC Security Target content (see next section), a SESIP5 Security Target must also include the rationale required for every SESIP evaluation:

- references to guidance documents describing the objectives for the environment,
- a reference to the public facing procedures describing how flaws are reported, tracked and corrected, and the resulting updates communicated,
- in the case of a composite platform, a description of the way the environment objectives are met for every platform part used in the composite platform.

The SESIP-specific elements of the Security Target may be included in a dedicated document and evaluated separately, preferably by the same laboratory, under the responsibility of a SESIP scheme's certification authority.

4.5.2 Assurance Components

Table 4-5: SESIP5 Assurance Requirements

Assurance Class	Assurance Families
ASE: Security Target evaluation	ASE_INT.1 – ST Introduction *ASE_OBJ.1 – Security requirements for the operational environment* **ASE_REQ.3 – Listed security requirements** *ASE_TSS.1 – TOE summary specification*
ADV: Development	ADV_ARC.1 Security architecture description ADV_FSP_4 – Complete functional specification ADV_TDS.3 Basic modular design ADV_IMP.2 Complete mapping of the implementation representation of the TSF
AGD: Guidance documents	AGD_OPE.1 – Operational user guidance AGD_PRE.1 – Preparative procedures
ALC: Life-cycle support	ALC_CMC.4 Production support, acceptance procedures and automation ALC_CMS.4 Problem tracking CM coverage ALC_DEL.1 Delivery procedures ALC_DVS.2 Sufficiency of security measures ALC_FLR.2 – Flaw reporting procedures ALC_TAT.1 Well-defined development tools
ATE: Tests	ATE_COV.1 Evidence of coverage ATE_DPT.1 Testing: basic design ATE_FUN.1 Functional testing ATE_IND.1 – Independent testing: conformance
AVA: Vulnerability Assessment	AVA_VAN.5 – Advanced methodical vulnerability analysis

All requirements but those in the ASE assurance class are as per [CC Part 3], and they are performed in the context of the prerequisite Common Criteria certification.

4.5.3 Security Target Requirements

As per section 4.3.3.

Notes:

The Security Target introduction (ASE_TSS.1) must refer to the prerequisite Common Criteria certification, in particular by including the certification body and certificate number.

In order to achieve the accessibility and reusability objectives, SESIP SFRs must be listed in the Security Target, and the associated compliance rationale must include a reference to the SFRs included in the original CC Security Target.

The TOE summary specification must be expressed as in other SESIP levels, typically “embedded” in the list of SFRs. Developers can reuse the content provided in the original CC Security Target, provided that they ensure that this content meets SESIP’s accessibility requirements.

ASE_OBJ.1 Security objectives for the operational environment

As per section 4.3.3.

ASE_REQ.3 Listed security requirements

As per section 4.3.3.

Note:

In order to achieve the accessibility and reusability objectives, SESIP SFRs must be listed in the Security Target, and the associated compliance rationale must include a reference to the SFRs included in the original CC Security Target.

ASE_TSS.1 TOE summary specification

As per section 4.3.3.

Note:

The TOE summary specification must be expressed as in other SESIP levels, typically “embedded” in the list of SFRs. Developers can reuse the content provided in the original CC Security Target, provided that they ensure that this content meets SESIP’s accessibility requirements.

4.5.4 Development Requirements

As per [CC Part 3] for all ADV requirements listed in Table 4-5.

4.5.5 Guidance Documents Requirements

As per [CC Part 3] for all AGD requirements listed in Table 4-5.

4.5.6 Life-cycle Support Requirements

As per [CC Part 3] for all ALC requirements listed in Table 4-5.

4.5.7 Tests Requirements

As per [CC Part 3] for all ATE requirements listed in Table 4-5.

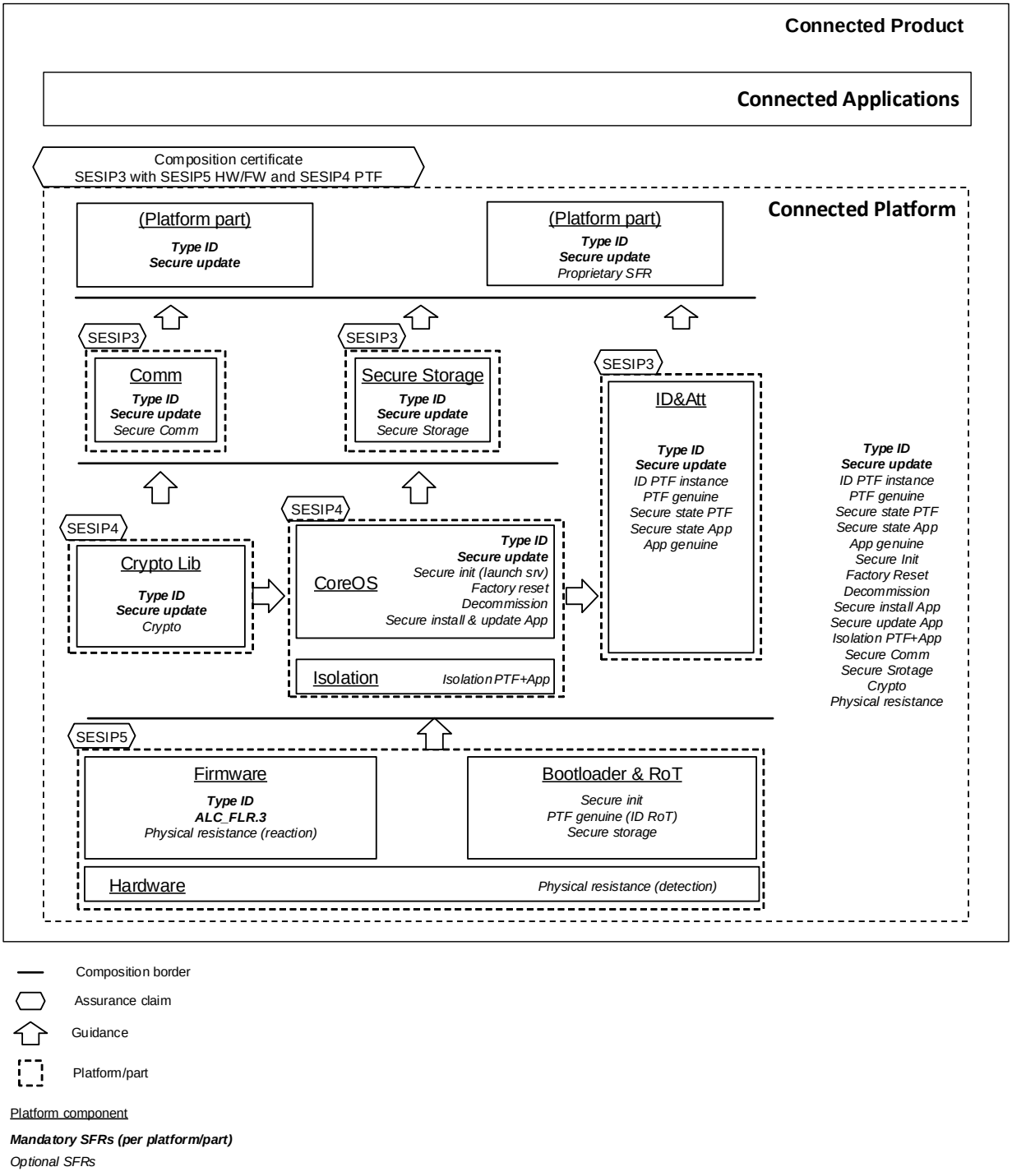
4.5.8 Vulnerability Assessment Requirements

As per [CC Part 3] for all AVA requirements listed in Table 4-5.

Annex A SESIP Evaluation Case Example

The figure provides an overall view of main SESIP principles showing example of SFRs claims, compositions and multi-assurance.

Figure A-1: SESIP Main Principles through an Example



Annex B Guidance: Attack Potential Rating

SESIP defines a methodology for evaluation as a variant to the Common Criteria standard. The definition of scheme-optimized attack potential and attack rating methodology is an essential part of any certification scheme based on SESIP. This appendix contains the standard methodology such a scheme-optimized methodology should be compatible with.

B.1 Principles

B.1.1 Identification and Exploitation Phases

The vulnerability analysis is performed in accordance with the Common Criteria ([ISO 15408-3]). The rating is performed in accordance with the [APSC] or [ISO 15408-3] v2.x methodology: by distinguishing in an "identification" and "exploitation" phase.

In the identification phase, the attacker has remote and physical (local) access to its own platform to identify the vulnerabilities and prepare the exploitation of that vulnerability to break one of the SFRs. In the exploitation phase, the attacker has at least remote access to a victim platform.

This base threat model covers scalable attacks over "the internet". For example, an attacker in the base threat model of SESIP might read out the firmware of the its own platform using an open debug interface, analyze it, and find a buffer overflow exploitable remotely on another instance of the platform.

B.1.2 Physical (local) Attacks and Remote Attacks

"Physical (local) attack" is any attack that requires an access undetected in its operational environment to the platform to exploit the vulnerability. For example: the (ab)use of a debug interface like JTAG, the reading out of an external memory like a flash memory, many of the fault injection and side channel attacks, physical modifications like a FIB, any software attacks applied through physical interfaces; but also, contactless fault injection or side channel attacks on a device but not visible to its owner.

"Remote attack" is any attack that does not require physical access to the platform to exploit the vulnerability. Note that attacks from a local network are considered to be remote attacks: SESIP does not distinguish between a trusted local network and the hostile internet, any connectivity made available is considered to be accessible to even the basic attacker. Nevertheless, the objectives for the environment may be used to describe the security context in which the product is supposed to be deployed, making some of the local attacks unpractical in that context.

B.2 Attack Potential Rating

The base reference for attack potential rating is the rating used in smart card devices as described in the most recent of version [APSC]. For illustration, the table below reproduces the version 3.0, April 2019 of [APSC]:

Table B-1: Attacks Rating

Factors	Identification	Exploitation	Notes
Elapsed time			
< one hour	0	0	-
< one day	1	3	
< one week	2	4	
< one month	3	6	
> one month	5	8	
Not practical	*	*	
Expertise			
Layman	0	0	-
Proficient	2	2	
Expert	5	4	
Multiple Expert	7	6	
Knowledge of the TOE			
Public	0	0	Critical or higher can only be claimed if all sites with access to that information are included in the scope of the evaluation at ALC_DVS.2 level (i.e. SESIP5).[1]
Restricted	2	2	
Sensitive	4	3	
Critical	6	5	
Very critical hardware design	9	NA	
Access to TOE			
< 10 samples	0	0	
< 30 samples	1	2	
< 100 samples	2	4	
> 100 samples	3	6	
Not practical	*	*	
Equipment			
None	0	0	
Standard	1	2	
Specialized	3	4	
Bespoke	5	6	
Multiple Bespoke	7	8	
Open samples			
Public	0	NA	Sensitive or higher can only be claimed if all sites with access to such open samples are included in the scope of the evaluation at ALC_DVS.2 level (i.e. SESIP5).[2]
Restricted	2	NA	
Sensitive	4	NA	
Critical	6	NA	

Certification schemes may adapt this table to fit with some technology's specificities (e.g. pure software target); in such case, all evaluations of platform (parts) based on this technology or equivalent must use the same updated table.

[1] Licensed evaluation labs and the CB should be considered to meet this requirement.

[2] Licensed evaluation labs should be considered to meet this requirement.

The table below defines the SESIP attack potential resistance met depending of attacks rating and required for each SESIP level.

Table B-2: Attack Potential Resistance Rating

SESIP level	Attack Ratings Allowed	SESIP Attack Potential Resistance
SESIP Assurance Level 1 (SESIP1)	0 – 15	Basic (AVA_VAN.1)
SESIP Assurance Level 2 (SESIP2)	16 – 20	Basic (AVA_VAN.2)
SESIP Assurance Level 3 (SESIP3)	21 – 24	Enhanced-Basic (AVA_VAN.3)
SESIP Assurance Level 4 (SESIP4)	25 – 30	Moderate (AVA_VAN.4)
SESIP Assurance Level 5 (SESIP5)	31 and above	High (AVA_VAN.5)

Annex C Example Use Cases

C.1 Generic Examples

C.1.1 IoT Cloud Connectivity Platform

The first example is an IoT cloud connectivity platform, which includes both hardware (a PCB with a microprocessor, some memory, and some peripherals, including at least a network connection) and software (an operating system, and a connectivity layer that manages all communications with the IoT cloud, and provides a high-level API to applications).

This is a complete IoT platform, which is likely to support most of the SFRs.

For instance, if the platform supports platform- and application-level attestations, it would include the following SFRs:

SFR	***Rationale for inclusion in ST***
Verification of Platform Identity: The platform provides a unique identification of the platform, including all its parts and their versions.	*This identification is mandatory for all SESIP platforms*
Verification of Platform Instance Identity: The platform provides a unique identification of that specific instantiation of the platform, including all its parts and their versions.	*An individual ID of each instance is needed to support attestations*
Attestation of Application Genuineness: The platform provides an attestation of the "Verification of Platform Identity" and "Verification of Platform Instance Identity", in a way that cannot be cloned or changed without detection.	*The platform can generate a proof of its identity, typically by using an instance-specific secret*
Secure Initialization of Platform: The platform ensures its authenticity and integrity during the platform initialization. If the platform authenticity or integrity cannot be ensured, the platform will go to a <*list of controlled states*>.	*This is the secure boot of the platform*
Attestation of Platform State: The platform provides an attestation of the state of the platform, such that it can be determined that the platform is in a known state.	*An attestation can be generated to provide some assurance about the platform's authenticity and current state*
Attestation of Application Genuineness: The platform provides an attestation of the application, in a way that cannot be cloned or changed without detection.	*This is an extension of the platform's identity proof and secure boot to its application*
Attestation of Application State: The platform provides an attestation of state of the application.	*The attestation provides assurance on the application state as seen by the platform.*

Such an IoT platform would most likely also cover the full range of life cycle management and update SFRs, not shown here, as well as the secure communications SFRs below:

SFR	*Rationale for inclusion in ST*
Secure Communication Support: The secure communication channel authenticates **the IoT Cloud endpoint** and protects against **disclosure, modification, replay, erasure** of messages between the endpoints, using **TLS1.2 with TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8 cipher suite**.	*The first step is to define what protocols are used to protect a given connection. Here, the connection to the cloud is protected by TLS1.2.*
Secure Communication Enforcement: The platform ensures the application can only communicate with **the IoT Cloud endpoint** over the secure communication channel(s) supported by the platform.	*The second step makes that secure protocol the only way to communicate with the IoT Cloud.*

Next comes the attack resistance. If we assume that the PP will be used to target platforms used in exposed and sensitive products, some extra resistance is required. Note that some of these SFRs may be instantiated more than once:

SFR	*Rationale for inclusion in ST*
Physical Attacker Resistance: The platform detects or prevents attacks by an attacker with physical access before the attacker compromises any of the other functional requirements, ensuring that the other functional requirements are not compromised.	*Physical attacks are fully covered, using any means available at the expected assurance level, targeting any part of the platform.*
Software Attacker Resistance: Isolation of Platform: 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 the other functional requirements.	*This level requires a separation between the platform and application, mostly through user/system distinctions expected on a complex chip.*
Software Attacker Resistance: Isolation of Application Parts: The platform provides isolation between parts of the application, such that an attacker able to run code as one of the **applications running in the main CPU's non-secure environment** cannot compromise the integrity and confidentiality of the other application parts.	*This is the isolation of a dedicated security subsystem and its critical assets (keys, in particular).*
Software Attacker Resistance: Isolation of Platform Parts: The platform provides isolation between platform parts, such that an attacker able to run code in **the main CPU's non-secure environment, including the application,** can compromise neither the integrity and confidentiality of **the main CPU's secure environment** nor the provision of any other security functional requirements.	*This is the isolation provided by a TEE between a secure and a non-secure world.*

Finally, an application shall provide services, which, for basic IoT applications, could be limited to high-level functions:

SFR	*Rationale for inclusion in ST*
Secure Encrypted Storage: The platform ensures that all data stored by the application, with the exception of **data stored in containers tagged as PLAIN_TEXT by the application**, is encrypted as specified in **AES (NIST FIPS 197)** with **a platform instance unique key of length 256 bits**.	*The platform offers two types of containers to the application, one of them being protected by encryption.*
Residual Information Purging: The platform ensures that all memory allocated through the platform, with **no exception**, is erased using the method specified in **NIST SP-800-88 Rev. 1 (zeroization)** before the memory is (re)used by the platform or application again and before an attacker can access it.	*All memory is cleared before reused (including RAM, as long as it is managed by the platform).*

C.1.2 Root-of-Trust Based on a Microcontroller

As security hardware becomes increasingly complex, the Root-of-Trust becomes a significant abstraction layer, to model the security functions provided by chip vendors. Arm has defined such an abstraction layer as part of their Platform Security Architecture (PSA) initiative, and they also have defined as part of the related PSA Certified initiative a Lightweight Protection Profile [PSA2 PP] for the certification of PSA-compliant roots-of-trust.

This Protection Profile can be mapped to SESIP, and it would lead to the following selection of SFRs:

SFR	*Rationale for inclusion in ST*
Verification of Platform Identity: The platform provides a unique identification of the platform, including all its parts and their versions.	*This SFR is mandatory for all SESIP platform (parts).*
Verification of Platform Instance Identity: The platform provides a unique identification of that specific instantiation of the platform, including all its parts and their versions.	*Dependency from "Attestation of Platform Genuineness" that provides a unique identifier*
Attestation of Platform Genuineness: The platform provides an attestation of the "Verification of Platform Identity" and "Verification of Platform Instance Identity", in a way that cannot be cloned or changed without detection.	*Corresponds to F.ATTESTATION for the platform's identity*
Secure Initialization of Platform: The platform ensures its authenticity and integrity during the platform initialization. If the platform authenticity or integrity cannot be ensured, the platform will go to a <*list of controlled states*>.	*Corresponds to F.INITIALIZATION and F.SECURE_STATE, providing secure boot features*
Attestation of Platform State: The platform provides an attestation of the state of the platform, such that it can be determined that the platform is in a known state.	*Corresponds to F.ATTESTATION and F.SECURE_STATE to provide an externally verifiable platform-level attestation, essential in PSA*

SFR	*Rationale for inclusion in ST*
Attestation of Application Genuineness: The platform provides an attestation of the application, in a way that cannot be cloned or changed without detection.	*Dependency from "Attestation of Application State", a basis for the application-level attestation*
Attestation of Application State: The platform provides an attestation of state of the application.	*Corresponds to F.ATTESTATION, here extended to include the application and its state (to be detailed in ST)*
Secure Update of Platform: The platform can be updated to a newer version in the field such that the integrity, authenticity and confidentiality of the platform is maintained.	*Corresponds to F.FIRMWARE_UPDATE for the platform part*
Secure Update of Application: The application can be updated to a newer version in the field such that the integrity, authenticity and confidentiality of the application is maintained.	*Corresponds to F.FIRMWARE_UPDATE for the application part*
Software Attacker Resistance: Isolation of Platform Parts: The platform provides isolation between platform parts, such that an attacker able to run code in **the NSPE** can compromise neither the integrity and confidentiality of **the SPE** nor the provision of any other security functional requirements.	*Corresponds to F.SOFTWARE_ISOLATION for the isolation between SPE and NSPE (both parts of a full IoT platform) at isolation Level 2*
Software Attacker Resistance: Isolation of Platform Parts: The platform provides isolation between platform parts, such that an attacker able to run code in **the NSPE's Application Roots-of-Trust** can compromise neither the integrity and confidentiality of **the PSA Root-of-Trust** nor the provision of any other security functional requirements.	*Corresponds to F.SOFTWARE_ISOLATION for the isolation the PSA RoT from Application RoTs (both parts of the NSPE) at isolation Level 3*
Cryptographic Operation: The platform provides the application with <*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*>.	*Corresponds to F.CRYPTO, to be iterated as much as needed*
Cryptographic Key Generation: The platform provides the application with a way to generate cryptographic keys for use in <*list of cryptographic algorithms*> as specified in <*specification*> for key lengths <*list of key lengths*>.	*Corresponds to F.CRYPTO, to be iterated as much as needed*
Cryptographic KeyStore: The platform provides the application with a way to store <*list of assets, such as cryptographic keys and passwords*> such that not even the application can compromise the <*authenticity, integrity, confidentiality*> of this data. This data can be used for the cryptographic operations <*list of operations*>.	*Corresponds to F.SECURE_STORAGE, which does not differentiate between crypto assets and general assets*

SFR	***Rationale for inclusion in ST***
Cryptographic Random Number Generation: The platform provides the application with a way based on <*list of entropy sources*> to generate random numbers to as specified in <*specification*>.	*Not explicitly mentioned in F.CRYPTO, but a likely implicit requirement*
Secure Encrypted Storage: The platform ensures that all data stored by the application, with the exception of <*list of data stored in plaintext*>, is encrypted as specified in <*specification*> with a platform instance unique key of key length <*key length*>.	*Corresponds to F.SECURE_STORAGE for application assets*
Audit Log Generation and Storage: The platform generates and maintains an audit log of <*list of significant security events*> and allows access and analysis of these logs following a specific <*access control policy*>.	*Corresponds to F.AUDIT*
Secure Debugging: The platform only provides <*list of endpoints*> authenticated as specified in <*specification*> with debug functionality. The platform ensures that all data stored by the application, with the exception of <*exceptions*>, is made unavailable.	*Corresponds to F.DEBUG with a few additional details about data protection*

C.2 Examples for Specific Use Cases

A scheme based on the SESIP standard should provide a way for platform developers (hardware and software) to show that certain security functional requirements are met, such that product developers can build a secure product on top and with it.

C.2.1 Secure Update of a Product (OTA)

If the platform developer wants to facilitate some secure update of the whole product (for example over the air update of critical part of a car), this would be the set of requirements for the platform:

SFR	***Rationale for inclusion in ST***
Verification of Platform Identity: The platform provides a unique identification of the platform, including all its parts and their versions.	*This SFR is mandatory for all SESIP platform (parts), and this is also required for taking care of versioning.*
Secure Initialization of Platform: The platform ensures its authenticity and integrity during the platform initialization. If the platform authenticity or integrity cannot be ensured, the platform will go to a <*list of controlled states*>.	*A secure boot mechanism is typically a prerequisite for guaranteeing authenticity*
Secure Update of Platform: The platform can be updated in the field such that the integrity, authenticity and confidentiality of the platform is maintained.	*This is the platform part of the update.*
Secure Update of Application: The product can be updated in the field such that the integrity, authenticity and confidentiality of the product is maintained.	*This is the application part of the update*

With these minimal SFRs, all aspects of secure update are covered, including its authenticity checks and the protection against typical attacks like version downgrade attacks, as described in the SESIP SFR.

Nevertheless, secure update usually comes with additional security measures. A more extensive platform supporting checks that the product is genuine and operating correctly, would add:

SFR	***Rationale for inclusion in ST***
Verification of Platform Instance Identity: The platform provides a unique identification of that specific instantiation of the platform, including all its parts and their versions.	*Making sure that each product built on the platform has a unique identifier*
Attestation of Platform Genuineness: The platform provides an attestation of the “Verification of Platform Identity” and “Verification of Platform Instance Identity”, in a way that cannot be cloned or changed without detection.	*Making sure that the platform can prove its identity*
Attestation of Platform State: The platform provides an attestation of the state of the platform, such that it can be determined that the platform is in a <*list of controlled states*>.	*Providing a proof that the platform is operating correctly*
Attestation of Application Genuineness: The platform provides an attestation of the application, in a way that cannot be cloned or changed without detection.	*Making sure that the application is genuine*
Attestation of Application State: The platform provides an attestation of state of the application.	*Proving that the application is genuine (from the platform’s viewpoint)*

C.2.2 A Blood Glucose Measurement Product (DTSec)

A blood glucose measurement product uses a platform for a secure channel over Bluetooth LE 4.1 with a mobile product, and cryptographic operations to make signatures and verify them.

This product will need to fulfill DTSec requirements, as defined in their Protection Profile [DTSec PP].

A minimal ST for the platform built from hardware and software could contain only the following security functional requirements:

SFR	***Rationale for inclusion in ST***
Verification of Platform Identity: The platform provides a unique identification of the platform, including all its parts and their versions.	*This SFR is mandatory for all SESIP platform (parts), and this is also required for taking care of versioning.*
Secure Communication Support: The platform provides the product with the secure communication channels it can optionally use. The channels authenticate the platform and **any external party** to each other, protect against **disclosure, modification, replay and erasure** of messages between these endpoints, using **Bluetooth LE 4.1 with options x,y,z always enabled**.	*The secure communication link required by DTSec*
Cryptographic Operation: The platform provides the product with hashing, signing, and signature verification functionality as specified in <*DTSec compliant references here*> for key lengths of **1024-2048 bit** and modes <*list of modes*>	*The supported algorithms must include DTSec-supported algorithms*
Cryptographic Random Number Generation: The platform provides the product with a way based on combined physical noise and cryptographic computation to generate random numbers to as specified in <*DTSec compliant references here, for example FIPS-something section x.y*>.	*Random number generation is another feature that must be provided by the platform.*

Such a minimal ST would be sufficient, but would leave significant tasks to the developer in order to meet the DTSec requirements

If the platform developer wants to facilitate the product developer to meet the DTSec scheme specific requirements easily, a hardened platform could implement parts of the requirements already in a way not even the product can circumvent. The DTSec evaluation would then be limited to verifying that “Attestation of Application Genuineness” and “Attestation of Application State” return the right values, and that the signature setting and verification is implemented in the way intended. For such a platform, the following SFRs could be used:

SFR	***Rationale for inclusion in ST***
Verification of Platform Identity: The platform provides a unique identification of the platform, including all its parts and their versions.	*This SFR is mandatory for all SESIP platform (parts), and this is also required for taking care of versioning.*

SFR	*Rationale for inclusion in ST*
Verification of Platform Instance Identity: The platform provides a unique identification of that specific instantiation of the platform, including all its parts and their versions	*Providing a unique identity to each product*
Attestation of Platform Genuineness: The platform provides an attestation of the "Verification of Platform Identity" and "Verification of Platform Instance Identity", in a way that cannot be cloned or changed without detection.	*Allowing each product to prove its identity*
Secure Initialization of Platform: The platform ensures its authenticity and integrity during the platform initialization. If the platform authenticity or integrity cannot be ensured, the platform will go to a <*list of controlled states*>.	*Providing secure boot to the product*
Attestation of Application Genuineness: The platform provides an attestation of the application, in a way that cannot be cloned or changed without detection.	*Also ensuring that the platform*
Secure Update of Platform: The platform can be updated in the field such that the integrity, authenticity and confidentiality of the platform is maintained.	*Making sure that the platform can be securely updated (mandatory in SESIP).*
Secure Update of Application: The product can be updated in the field such that the integrity, authenticity and confidentiality of the product is maintained.	*Extending the update feature to the application*
Secure Communication Support: The platform provides the product with the secure communication channels it can optionally use. The channels authenticate the platform and **any external party** to each other, protect against **disclosure, modification, replay and erasure** of messages between these endpoints, using **Bluetooth LE 4.1 with options x,y,z always enabled**.	*The secure communication link required by DTSec*
Secure Communication Enforcement: The platform ensures the application can only communicate with **any external party** over the secure communication channel(s) supported by the platform.	*With this SFR, the application is forced to use the provided secure communication link*
Software Attacker Resistance: Isolation of Platform: 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 the other functional requirements.	*The application cannot interfere with the platform.*
Cryptographic Operation: The platform provides the product with hashing, signing, and signature verification functionality as specified in <*DTSec compliant references here*> for key lengths of **1024-2048 bit** and modes <*list of modes*>.	*The supported algorithms must include DTSec-supported algorithms*
Cryptographic Random Number Generation: The platform provides the product with a way based on combined physical noise and cryptographic computation to generate random numbers to as specified in <*DTSec compliant references here, for example FIPS-something section x.y*>.	*Random number generation is another feature that must be provided by the platform.*

SFR	***Rationale for inclusion in ST***
Secure Encrypted Storage: The platform ensures that all data stored by the product, with the exception of **data stored in attached SD cards**, is encrypted as specified in <*DTSEC compliant references here, for example AES FIPS-something section x.y*> with a product unique key of key length of **128 bits**.	*This adds a functional layer to help the application in the implementation of secure storage*

Annex D Security Target Template

This annex defines a template for a SESIP security target. In this version of the standard, this template applies only for levels SESIP1 to SESIP3, for which full evaluations can be performed.

Apart from the section numbering, the template provides the various required sections in a SESIP Security Target. The descriptions provided *<in italics between brackets>* need to be replaced with the Security Target's actual descriptions.

Notes may be added where required using that specific format, which are not expected to be included in actual Security Targets.

D.1 Security Target Title Page

The following information is required in the Security target Title Page

Security Target for *<Platform name>*

Version *<1.0>*, dated *<yyyy-mm-dd>*

<Development Organization>

Based on SESIP methodology, version *<SESIP Version>*

D.2 Introduction

The Security Target describes the platform (in this chapter) and the exact security properties of the platform that are evaluated against SESIP.

D.2.1 ST Reference

See title page.

D.2.2 Platform Reference

<Unique identification of the platform>

TOE name	*<TOE name>*
TOE version	*<TOE version>*
TOE identification	*<TOE id details>*
TOE Type	*<e.g. microcontroller platform for IoT applications>*

D.2.3 Included Guidance Documents

The following documents are included with the platform:

Reference	Name	Version
<[Ref]>	*<Full title of the document>*	*<Vx.y>*

The guidance should list in particular all the documents that will be provided to the valuator for the documentation review, covering AGD_OPE.1 and AGD_PRE.1. This documentation is expected to be available to the customers without restrictions.

D.2.4 (Optional) Other Certification

The product has previously been evaluated following the *<name/reference of the methodology>*:

Scheme	*<Scheme name>*
Certification body	*<Name of certification body>*
Certification number	*<Full certificate number>*
Certificate date	*<Certificate issuance date>*

The developer may also include a link to the certificate on internet, and/or a copy of the certificate.

D.2.5 Platform Functional Overview and Description

<A short introduction and description of the platform shall be provided. Typically, this would be taken from the datasheet. An overview picture and feature set should be described. 1-2 pages are expected.>

The TOE consists of a *<describe TOE parts; e.g. microcontroller and platform implementing secure boot>*.

The TOE is intended to be used by *<e.g. an integrator that deploys it into an IoT solution together with its own user application, providing assurance that the IoT application is securely booted and operates securely>*.

The main security features of the TOE are as follows:

- *<Secure boot.*
- *Crypto processor.*
- *….>*

The TOE scope is depicted in *<Refer to a figure showing the TOE architecture and scope>*. The blue parts are within the evaluation scope and the gray parts are outside of the evaluated scope. The out of scope part comprises *<to be completed by developer>*.

The physical scope includes *<write specific scope details, which may be a silicon chip, a PCB, …>*

D.3 Security Objectives for the Operational Environment

D.3.1 Platform Objectives for the Operational Environment

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

<List all mandatory objectives for the environment with reference to where in the guidance documents this objective is described. Examples:

- *The application shall verify the correct version of all platform components it depends on (as described in [Manual] section "version check").*
- *The application should support the invocation of the update mechanism as described in [Manual] section "updates".*
- *The platform shall only be deployed in environments where there is no physical attacker possible (as described in [Datasheet] section "limitations").*
- *The application shall not allow execution of hostile code (as described in [Datasheet] section "limitations").*

>

The description of the security objectives for the environment shall include enough information for the platform customer (product vendor/integrator) to understand the requirements and implement them. In particular, every security objective shall include a precise reference to the user guidance.

D.3.2 Inherited Objectives for the Operational Environment

This section is only required for composite platforms that include platform parts that have been evaluated under a SESIP scheme. A dedicated subsection must be included for every platform part.

The platform includes platform parts that have previously been evaluated under <*some SESIP scheme*>. That platform part defined objectives for its own operational environment, which have been handled by the platform as follows:

<For every operational environment objective defined in a platform part, recall the objective and explain how this objective has been handled by the platform:

- *If the operational objective is entirely or partly covered by the platform, then provide an explanation by referring to the platform's SFRs.*
- *If the operational objective is not or partly covered by the platform, then provide an explanation by referring to the platform's own objectives for the operational environment.*

>

D.4 Security Requirements and Implementation

D.4.1 Security Assurance Requirements

The claimed assurance requirements package is <*Select one from SESIP1, SESIP2, SESIP3, SESIP4, SESIP5*> as defined in Chapter 4.

Assurance requirements are typically simply listed in a Security Target, but in SESIP, and in particular in the lowest levels, where limited evidence is provided, some basic evidence needs to be included in the ST. This applies at least for ALC_FLR.2 (at all levels) and for AVA_VAN.1 (for SESIP1).

The following two Security Assurance Requirements are provided as examples.

Flaw Reporting Procedure (ALC_FLR.2)

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

<*Describe the procedure, including where flaws 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.*>

Vulnerability Survey (AVA_VAN.1)

In accordance with the requirement for a vulnerability analysis survey (AVA_VAN.1) the developer has performed a vulnerability survey and submits the following test results to demonstrate the consideration of publicized potential vulnerabilities relating to the TOE:

<*Describe what form the vulnerability survey took and describe the testing performed as a result, e.g. considered all attacks relating to a basic attack potential are considered in X public domain tool and so executed the tool with specified results.*>

D.4.2 Security Functional Requirements

The platform fulfills the following Security Functional Requirements:

The Identification of platform and the Secure update of platform requirements are explicitly listed here, because they are mandatory in all SESIP Security Targets. Additional SFRs are then added to suit the vendor's objectives.

For every SFR, a description of the implementation proposed in the TOE and of the way this implementation is assessed also needs to be included

Verification of Platform Identity

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

Conformance rationale:

<*Description of how the platform implements a way to tell that something it the certified platform. Expected is a short paragraph like:*

The platform has a dedicated memory area <name> that includes a <#bits>-bit identifier unique for this platform series, encoding the "Platform reference" into the value 0x12345678.>

<Description of how this implementation is assessed, for instance by reference to testing, conformance to another standard, or reliance on other parties. Expected is a short paragraph like:

The <name> memory area is written as part of the production process, and the production testing procedures verify the value has been written correctly.>

Secure update of platform

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

Conformance rationale:

<Description of how the platform implements the update. Expected is 1-3 small paragraphs like:

The platform has a secure update mechanism named <name>. This mechanism can be used by the application to initiate an update. Updates are checked for integrity and authenticity by verifying a <#bits>-bits <signature algorithm> signature against the stored public key of the platform developer. The updates are encrypted for the platform series, protecting the confidentiality of the platform and any data added to the update.

The update mechanism verifies prior to installation that the version of the update is higher (more recent) than the current version installed, preventing roll-back and downgrading attacks.

The application developer is required to make the update mechanism available via its infrastructure, as described in section "Security Objectives for the operational environment".

>

<Description of how this implementation is assessed, for instance by reference to testing, conformance to another standard, or reliance on other parties. Expected is a short paragraph like:

The update mechanism is tested for robust functioning by <testing lab> and evaluated for security by <scheme/lab>.

>

<Security Functional Requirement (SFR) from SESIP>

<SFR text (from Chapter 3) with all lists and other optional items (identified in Chapter 3 by "<…>") filled out>

Conformance rationale:

<Description of how the platform implements the SFR>

<Description of how this implementation is assessed, for instance by reference to testing, conformance to another standard, or reliance on other parties. Expected is a short paragraph.>

D.4.3 Additional Security Functional Requirements

Any security functional requirement that is not defined in SESIP must be included in such a dedicated section (clearly separated from SESIP requirements), and this section must start with the following statement.

In addition to the standard SESIP security functional requirements listed above, the TOE includes some specific requirements that listed below. Note that these requirements may not be recognized by other stakeholders, since they are not part of the SESIP methodology.

<List the platform-specific SFRs, providing clear explanations>

<Security Functional Requirement (SFR)>

<SFR text written in a clear and understandable way that describes the security feature expected. The text may be supported by a short explanation providing detailed expectations.>

Conformance rationale:

<Description of how the platform implements the SFR>

<Description of how this implementation is assessed, for instance by reference to testing, conformance to another standard, or reliance on other parties. Expected is a short paragraph.>

D.5 Mapping and Sufficiency Rationales

The ST must also include a rationale that explains how the assurance requirements are addressed in the Security Target or planned to be addressed in the evaluation. The present section provides examples for levels SESIP1 to SESIP4 (since a SESIP5 ST is a standard CC ST), and the last column provides some comments and indications on the answer.

In an actual Security Target, only one of these rationales should be included, without the comments, corresponding to the selected SESIP level.

D.5.1 SESIP1 Sufficiency

SESIP1 deliverables, required to demonstrate good security practice, are mostly contained in the Security Target itself, complemented with the product's security documentation

Assurance Class	Assurance Families	Covered by	Rationale	Indications
ASE: Security Target evaluation	ASE_INT.1 ST Introduction	Section "Introduction" and title page	The ST reference is in the Title, the TOE reference in the "Platform Reference", the TOE overview and description in "Platform Functional Overview and Description".	*Generic answer*
	ASE_OBJ.1 Security requirements for the operational environment	Section "Security Objectives for the Operational Environment"	The objectives for the operational environment in "Security Objectives for the Operational Environment" refer to the guidance documents.	*Generic answer*

Assurance Class	Assurance Families	Covered by	Rationale	Indications
	ASE_REQ.3 Listed Security requirements	Section “Security Functional Requirements”	All SFRs in this ST are taken from SESIP. “Verification of Platform Identity” is included. “Secure Update of Platform” is included.	*Generic answer, including a reference to the two SFRs that must be included*
	ASE_TSS.1 TOE Summary Specification	Section “Security Requirements and Implementation”	All SFRs are listed per definition, and for each SFR the implementation and verification are defined in “Security Functional Requirements”.	*Generic answer*
AGD: Guidance documents	AGD_OPE.1 Operational user guidance	*<Description of which developer evidence is used to meet this requirement>*	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.	*Generic answer, referring to an external deliverable*
	AGD_PRE.1 Preparative procedures	*<Description of which developer evidence is used to meet this requirement>*	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.	*Generic answer, referring to an external deliverable*
ALC: Life-cycle support	ALC_FLR.2 Flaw reporting procedures	Section “Flaw reporting procedures (ALC_FLR.2)”	The flaw reporting and remediation procedure is described.	*A description of the procedure is required, here in the ST*
AVA_VAN.1	AVA_VAN.1 Vulnerability survey	Section “Vulnerability survey (AVA_VAN.1)”	The vulnerability survey and associated test results are described.	*A description of the survey is required, here in the ST*

D.5.2 SESIP2 Sufficiency

SESIP2 deliverables also add basic documentation required to perform a black-box evaluation

Assurance Class	Assurance Families	Covered by	Rationale	Indications
ASE: Security Target evaluation	ASE_INT.1 ST Introduction	Section "Introduction" and title page	The ST reference is in the Title, the TOE reference in the "Platform Reference", the TOE overview and description in "Platform Functional Overview and Description".	*From SESIP1*
	ASE_OBJ.1 Security requirements for the operational environment	Section "Security Objectives for the Operational Environment"	The objectives for the operational environment in "Security Objectives for the Operational Environment" refer to the guidance documents.	*From SESIP1*
	ASE_REQ.3 Listed Security requirements	Section "Security Functional Requirements"	All SFRs in this ST are taken from SESIP. "Verification of Platform Identity" is included. "Secure Update of Platform" is included.	*From SESIP1*
	ASE_TSS.1 TOE Summary Specification	Section "Security Requirements and Implementation"	All SFRs are listed per definition, and for each SFR the implementation and verification are defined in "Security Functional Requirements".	*From SESIP1*
ADV: Development	ADV_FSP.4 Complete functional specification	*<Description of which developer evidence is used to meet this requirement>*	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.	*Generic answer, referring to an external deliverable*

Assurance Class	Assurance Families	Covered by	Rationale	Indications
AGD: Guidance documents	AGD_OPE.1 Operational user guidance	*<Description of which developer evidence is used to meet this requirement>*	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.	*From SESIP1*
	AGD_PRE.1 Preparative procedures	*<Description of which developer evidence is used to meet this requirement>*	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.	*From SESIP1*
ALC: Life-cycle support	ALC_FLR.2 Flaw reporting procedures	Section “Flaw reporting procedures (ALC_FLR.2)”	The flaw reporting and remediation procedure is described.	*From SESIP1*
ATE: Tests	ATE_IND.1 Independent testing: conformance	*<Description of which developer evidence is used to meet this requirement>*	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.	*To be instantiated with a list of deliverables*
AVA_VAN.2	AVA_VAN.2 Vulnerability analysis	N.A. A vulnerability analysis is performed by the platform evaluator to ascertain the presence of potential vulnerabilities.	The platform evaluator performs penetration testing, to confirm that the potential vulnerabilities cannot be exploited in the operational environment for the TOE. Penetration testing is performed by the platform evaluator assuming an attack potential of Basic.	*Generic answer, referring to a platform evaluator task*

D.5.3 SESIP3 Sufficiency

SESIP3 deliverables also add basic documentation required to perform a white-box evaluation, as well as basic evidence of the use of configuration management.

Assurance Class	Assurance Families	Covered by	Rationale	Indications
ASE: Security Target evaluation	ASE_INT.1 ST Introduction	Section "Introduction" and title page	The ST reference is in the Title, the TOE reference in the "Platform Reference", the TOE overview and description in "Platform Functional Overview and Description".	*From SESIP1*
	ASE_OBJ.1 Security requirements for the operational environment	Section "Security Objectives for the Operational Environment"	The objectives for the operational environment in "Security Objectives for the Operational Environment" refer to the guidance documents.	*From SESIP1*
	ASE_REQ.3 Listed Security requirements	Section "Security Functional Requirements"	All SFRs in this ST are taken from SESIP. "Verification of Platform Identity" is included. "Secure Update of Platform" is included.	*From SESIP1*
	ASE_TSS.1 TOE Summary Specification	Section "Security Requirements and Implementation"	All SFRs are listed per definition, and for each SFR the implementation and verification are defined in "Security Functional Requirements".	*From SESIP1*
ADV: Development	ADV_FSP.4 Complete functional specification	*<Description of which developer evidence is used to meet this requirement>*	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.	*From SESIP2*

Assurance Class	Assurance Families	Covered by	Rationale	Indications
	ADV_IMP.3 Complete mapping of the implementation representation of the TSF to the SFRs	*<Description of which developer evidence is used to meet this requirement>*	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.	*To be instantiated with a list of deliverables (typically, annotated source)*
AGD: Guidance documents	AGD_OPE.1 Operational user guidance	*<Description of which developer evidence is used to meet this requirement>*	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.	*From SESIP1*
	AGD_PRE.1 Preparative procedures	*<Description of which developer evidence is used to meet this requirement>*	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.	*From SESIP1*
ALC: Life-cycle support	ALC_CMC.1 Labelling of the TOE	*<Description of which developer evidence is used to meet this requirement>*	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.	*To be instantiated with a list of deliverables*
	ALC_CMS.1 TOE CM Coverage	*<Description of which developer evidence is used to meet this requirement>*	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.	*To be instantiated with a list of deliverables*
	ALC_FLR.2 Flaw reporting procedures	Section “Flaw reporting procedures (ALC_FLR.2)”	The flaw reporting and remediation procedure is described.	*From SESIP1*
ATE: Tests	ATE_IND.1 Independent testing: conformance	*<Description of which developer evidence is used to meet this requirement>*	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.	*From SESIP2*

Assurance Class	Assurance Families	Covered by	Rationale	Indications
AVA_VAN.3	AVA_VAN.3 Focused Vulnerability analysis	N.A. A vulnerability analysis is performed by the platform evaluator to ascertain the presence of potential vulnerabilities.	The platform evaluator performs penetration testing, to confirm that the potential vulnerabilities cannot be exploited in the operational environment for the TOE. Penetration testing is performed by the platform evaluator assuming an attack potential of Enhanced-Basic.	*Generic answer, referring to a platform evaluator task, similar to SESIP2, but considering more skilled attackers*

D.5.4 SESIP4 Sufficiency

SESIP4 deliverables mostly add additional evidence of good development practice and testing sufficiency. In most cases, the rationale used for SESIP4 will reference the associated CC requirement on which the SESIP4 evaluation is built.

Assurance Class	Assurance Families	Covered by	Rationale	Indications
ASE: Security Target evaluation	ASE_INT.1 ST Introduction	Section “Introduction” and title page	The ST reference is in the Title, the TOE reference in the “Platform Reference”, the TOE overview and description in “Platform Functional Overview and Description”.	*From SESIP1.* *No need to add anything to cover the required reference to the previous CC certificate*
	ASE_OBJ.1 Security requirements for the operational environment	Section “Security Objectives for the Operational Environment”	The objectives for the operational environment in “Security Objectives for the Operational Environment” refer to the guidance documents.	*From SESIP1*

Assurance Class	Assurance Families	Covered by	Rationale	Indications
	ASE_REQ.3 Listed Security requirements	Section "Security Functional Requirements"	All SFRs in this ST are taken from SESIP. "Verification of Platform Identity" is included. "Secure Update of Platform" is included. SFRs are mapped to CC SFRs from the associated CC Security Target.	*Adapted to include the reference to original CC SFRs*
	ASE_TSS.1 TOE Summary Specification	Section "Security Requirements and Implementation"	All SFRs are listed per definition, and for each SFR the implementation and verification are defined in "Security Functional Requirements".	*Adding a specific requirement to map*
ADV: Development	ADV_ARC.1 Security architecture description	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This is an AVA.VAN.4 dependency.*
	ADV_FSP.4 Complete functional specification	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This is an AVA.VAN.4 dependency.*
	ADV_IMP.3 Complete mapping of the implementation representation of the TSF to the SFRs	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet requirements ADV_TDS.3 and ADV_IMP.1 in the associated CC evaluation. This is considered equivalent to the ADV_IMP.3 requirement.	*Both ADV_TDS.3 and ADV_IMP.1 are AVA_VAN.4 dependencies*

Assurance Class	Assurance Families	Covered by	Rationale	Indications
AGD: Guidance documents	AGD_OPE.1 Operational user guidance	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This is an AVA.VAN.4 dependency.*
	AGD_PRE.1 Preparative procedures	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This is an AVA.VAN.4 dependency.*
ALC: Life-cycle support	ALC_CMC.1 Labelling of the TOE	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*A higher component (ALC_CMC.3) is required in any EAL4 evaluation.*
	ALC_CMS.1 TOE CM Coverage	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*A higher component (ALC_CMS.3) is required in any EAL4 evaluation.*
	ALC_FLR.2 Flaw reporting procedures	Section “Flaw reporting procedures (ALC_FLR.2)”	The flaw reporting and remediation procedure is described. The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*ALC_FLR.2 is not required in CC evaluations, but it must be included for any product that later intends to get a SESIP certificate.*
	ALC_DEL.1 Delivery procedures	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This component is required in any EAL4 evaluation*

Assurance Class	Assurance Families	Covered by	Rationale	Indications
	ALC_DVS.1 Identification of security measures	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This component is required in any EAL4 evaluation*
	ALC_TAT.1 Well-defined development tools	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This component is required in any EAL4 evaluation*
ATE: Tests	ATE_COV.1 Evidence of coverage	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*A higher component (ATE_COV.2) is required in any EAL4 evaluation*
	ATE_FUN.1 Functional testing	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This component is required in any EAL4 evaluation*
	ATE_IND.1 Independent testing: conformance	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*A higher component (ATE_IND.2) is required in any EAL4 evaluation*
AVA_VAN.4	AVA_VAN.4 Methodical Vulnerability analysis	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This is a SESIP4 requirement, so any CC evaluation performed under EAL5 must include an AVA_VAN.4 augmentation*

D.5.5 SESIP5 Sufficiency

SESIP5-specific deliverables only add additional evidence through the SESIP Security Target, although it covers other requirements such as ALC.FLR.2. All other requirements are expected to be trivially covered by the underlying CC evaluation.

Assurance Class	Assurance Families	Covered by	Rationale	Indications
ASE: Security Target evaluation	ASE_INT.1 ST introduction	Section "Introduction" and title page	The ST reference is in the Title, the TOE reference in the "Platform Reference", the TOE overview and description in "Platform Functional Overview and Description".	*From SESIP4*
	ASE_CCL.1 Conformance claims	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This component is required in any EAL4 evaluation*
	ASE_OBJ.1 Security requirements for the operational environment	Section "Security Objectives for the Operational Environment"	The objectives for the operational environment in "Security Objectives for the Operational Environment" refer to the guidance documents.	*From SESIP1*
	ASE_REQ.3 Listed security requirements	Section "Security Functional Requirements"	All SFRs in this ST are taken from SESIP. "Verification of Platform Identity" is included. "Secure Update of Platform" is included. SFRs are mapped to CC SFRs from the associated CC Security Target.	*From SESIP4*
	ASE_SPD.1 Security problem definition	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This component is required in any EAL4 evaluation*

Assurance Class	Assurance Families	Covered by	Rationale	Indications
	ASE_TSS.1 TOE Summary specification	Section "Security Requirements and Implementation"	All SFRs are listed per definition, and for each SFR the implementation and verification are defined in "Security Functional Requirements".	*Adding a specific requirement to map*
ADV: Development	ADV_ARC.1 Security architecture description	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This is an AVA.VAN.5 dependency.*
	ADV_FSP.4 Complete functional specification	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This is an AVA.VAN.5 dependency.*
	ADV_TDS.3 Basic modular design	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This is an AVA.VAN.5 dependency.*
	ADV_IMP.2 Complete mapping of the implementation representation of the TSF	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This is an AVA.VAN.5 dependency.*
AGD: Guidance documents	AGD_OPE.1 Operational user guidance	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This is an AVA.VAN.5 dependency.*

Assurance Class	Assurance Families	Covered by	Rationale	Indications
	AGD_PRE.1 Preparative procedures	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This is an AVA.VAN.5 dependency.*
ALC: Life-cycle support	ALC_CMC.4 Production support, acceptance procedures and automation	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This component is required in any EAL4 evaluation*
	ALC_CMS.4 Problem tracking CM Coverage	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This component is required in any EAL4 evaluation*
	ALC_DEL.1 Delivery procedures	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This component is required in any EAL4 evaluation*
	ALC_DVS.1 Identification of security measures	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This component is required in any EAL4 evaluation*
	ALC_FLR.2 Flaw reporting procedures	Section “Flaw reporting procedures (ALC_FLR.2)”	The flaw reporting and remediation procedure is described. The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*ALC_FLR.2 is not required in CC evaluations, but it must be included for any product that later intends to get a SESIP certificate.*

Assurance Class	Assurance Families	Covered by	Rationale	Indications
	ALC_TAT.1 Well-defined development tools	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This component is required in any EAL4 evaluation*
ATE: Tests	ATE_COV.1 Evidence of coverage	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*A higher component (ATE_COV.2) is required in any EAL4 evaluation*
	ATE_DPT.1 Testing: basic design	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This component is required in any EAL4 evaluation*
	ATE_FUN.1 Functional testing	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This component is required in any EAL4 evaluation*
	ATE_IND.1 Independent testing: conformance	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*A higher component (ATE_IND.2) is required in any EAL4 evaluation*
AVA_VAN.4	AVA_VAN.4 Methodical Vulnerability analysis	Refer to associate CC certificate.	The platform evaluator has determined that the provided evidence is suitable to meet the requirement in the associated CC evaluation.	*This is a SESIP4 requirement, so any CC evaluation performed under EAL5 must include an AVA_VAN.4 augmentation*