Security Evaluation Scheme for IoT Platforms

Version 1.2

Contents

1 Introduction

This document describes the Security Functional Requirements, Security Assurance Requirements and Governance for the Security Evaluation Scheme for IoT Platforms (SESIP).

Currently, this document is draft and for use in the V1.2 evaluations.

From the use of SESIP in evaluations we will learn and adjust to a newer, even better version. Newer versions are expected at least every quarter of a year during this initial period.

See "Further development of SESIP" for areas and aspects already identified. See the SESIP site < https://trustcb.com/iot/sesip/> for updated versions of this document and useful templates, profiles and other supporting documents.

1.1 Structure of this document

This document is structured as follows:

Section 1.2 defines a number of terms that are used in this document. Most definitions rely on, and assume knowledge of, definitions in the CC and CCRA.

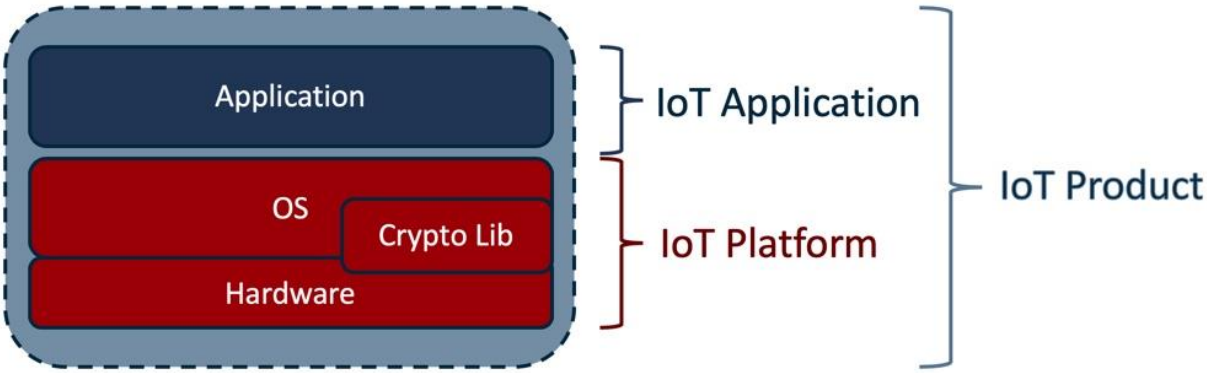
Chapter 2 defines the Security Functional Requirements. It completely replaces CC Part 2 for this standard. CC Part 2 is not used in SESIP.

Chapter 3 describes the Security Assurance Requirements. It defines five hierarchical levels of assurance for IoT platforms ("alternative EALs"). This chapter also defines additional assurance components that are used in these levels, but are not used in CC part 3.

Chapter 4 describes the governance structure of the scheme.

Chapter 5 provides a number of use-cases as worked examples.

1.2 Terms used in this document



An **IoT Platform** is the hardware/software providing an operating environment for an IoT Application. IoT Platforms parts can be developed and evaluated separately, for example by evaluating the cryptographic library, an OS, hardware, and then combining them. In terms of the Common Criteria, the IoT Platform (part) identified in the ST is our TOE.

An **IoT Application** is the software running on the IoT Platform adding domain-specific functionality. An IoT Platform together with an IoT Application in total form an **IoT Product**, providing the user with a complete functionality. From the platform point of view, there is only one IoT Application, even if

this IoT Application is separated in many different applications parts from the IoT Application developer point of view.

For brevity, "IoT Platform", "IoT Application" and "IoT Product" are abbreviated to "platform", "application" and "product" in this document.

Platform developers build platforms parts and supply these to composite developers. They will be able to market more effectively if they can show that their platform provides certain functionalities in a secure manner, so the composite developer can get cheaper and less risky evaluations from composite evaluators.

Platform evaluators and certifiers verify whether a platform is secure in accordance with the standard described in this document.

Composite developers build their own platform extensions or application upon an underlying platform. Composite developers of a product want to provide their users with assurance that the product is secure, and for this reason may submit their product to product-specific schemes.

Composite evaluators and certifiers of a platform verify whether a composed platform meets the requirements by reusing the underlying assurance. Composite evaluators and certifiers of a product verify if product is secure in accordance with their product-specific scheme. To save work, composite evaluators and certifiers wish to re-use evaluation results from any platforms that are used by the composite developer.

Users are end-users of an IoT product. Users want to know their product is secure in the way they expect for the product type, which at least includes confidentiality of their data. They probably neither know nor care what platform the product is built on.

Regulators want to make sure that IoT products are secure. They can therefore mandate that products are approved by a product-specific scheme or use secure platforms as certified under this scheme.

1.3 IoT use case and threat model

IoT is a broad term, but always contains a product ("thing") and some form of connectivity ("internet").

The minimum threat model has an attacker with access to the "internet" connection.

Platforms can expand the threat model considered with the use of the SFRs "Physical attacker resistance", "Software attacker resistance: isolation of platform", "Software attacker resistance: isolation of application parts", and "Software attacker resistance: isolation of platform parts".

Following the trials this section will be expanded to explicitly mention the assets/objectives and services that are visible at the platform definition.

2 Security Functional Requirements

IoT 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.

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

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.

Only the Security Functional Requirements listed in this chapter shall be used in Security Targets. Should you have a functionality that cannot be captured in the below Security Functional Requirements, contact the scheme owner TrustCB <SESIP@trustcb.com>. See also "Further development of SESIP".

Note that all Security Targets should include the "Identification of platform type" SFR, and all should either include the "Secure update of platform" SFR or argue under ALC_FLR.2 why updates are not applicable.

2.1 Identification and attestation of platforms and applications

Identification and attestation allows 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.

2.1.1 Identification of platform type

The platform provides a unique identification of the platform type, 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 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).

Considerations

The ST describes the platform (=TOE) scope, 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, must be the certified product.

This functional requirement is mandatory for all platforms, to ensure customers can verify that they have the correct certified platform.

2.1.2 Identification of individual platform

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 "Identification of platform type".

2.1.3 Genuine platform instantiation

The platform provides an attestation of the "Identification of platform type" and "Identification of individual platform", 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.

Users of the platform certificates of this scheme, 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 type and not a device 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 must be considered as part of the evaluation.

2.1.4 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 a secure state.

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.

2.1.5 Attested secure state of platform

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

Value

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

Consideration

Implies that "Genuine platform" and "Secure initialization of platform" is also implemented.

2.1.6 Genuine application

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 "Genuine platform" and "Secure initialization of platform" are also implemented.

2.1.7 Attested state of application

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

Implies that "Genuine application" and "Attested secure state of platform" are also implemented.

2.2 Product lifecycle: Factory reset / Install / Update / Decommission

The platform must 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.

2.2.1 Factory reset of platform

The platform can be reset to the state it was originally after production.

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. As all data is destroyed, the user's data is also destroyed.

Considerations

The platform must still be functional after the factory reset.

This reset must destroy all data (including the data of the application) received after production, 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 "Genuine platform instantiation" and "Attested secure state of platform", 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 must 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 must be appropriate for the persistent memory technology and attack potential. See also "Residual information purging".

2.2.2 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 disclosed during this installation.

Consideration

A platform offering this may consider also offering "Secure update of application".

2.2.3 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 must be explained in the ST under ALC_FLR.2.

The update mechanism must 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 must have a (globally) unique identifier as per "Identification of platform type", and may require their own (re-)evaluation and (re-)certification.

2.2.4 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 must 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.

A platform offering this may consider also offering "Secure install of application".

2.2.5 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 must not be functional after the decommissioning.

The decommissioning must destroy all data (including the data of the application) received after production. Decommissioning must also destroy all application parts not explicitly marked as exempt of decommissioning. Destruction must be such that the data and application parts can not be compromised even when the product is also physically accessible to an attacker.

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

If the application is (partially) defective, it must 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 must be appropriate for the persistent memory technology and attack potential. See also "Residual information purging".

2.3 Secure communication

Typically a product will use secure channels to communicate with a server, another IoT device 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

2.3.1 Secure communication support

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

The secure communication channel authenticates <which kind of endpoints> and protects against <disclosure, modification, replay, erasure> of messages between the endpoints, using <TLS 1.2 with … ciphersuites, IPSec with …, ….>.

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

If multiple different secure communication channels are provided, iterate this SFR.

2.3.2 Secure communication enforcement

The platform ensures the application can only communicate with <which kind of endpoints: for example all/remote/local endpoints> over the secure communication channel(s) supported by the platform.

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.

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 this lower communication.

2.4 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[1].

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.

2.4.1 Limited physical attacker resistance

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

Value

For situations where a physical attacker is in scope for a limited set of functionality such as valuable assets warranting a physical attack, but not for all functionality.

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 must 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 device 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 can be countered with SFRs "Identification of individual platform" together with "Genuine platform instantiation", allowing an application or user to detect replacement.

2.4.2 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 must now also be considered in the vulnerability analysis as accessible to the attacker (such as an "evil maid").

[1] That is: an attacker would have to "break" the application or platform first in order to install his code: there are no regular ways to do this.

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 device 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 can be countered with SFRs "Identification of individual platform" together with "Genuine platform instantiation", allowing an application or user to detect replacement.

2.4.3 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 an OS with kernel-user mode separation.

2.4.4 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 of other parts of the application code that are too complex to be show to be secure.

Considerations

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.

2.4.5 Software attacker resistance: isolation of platform parts

The platform provides isolation between <list platform parts>, such that an attacker able to run code in <list platform parts> can not compromise the integrity and confidentiality of <the other platform parts> or any other 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

A platform offering this functionality would typically also claim "Introduction".

This typically would require a micro-kernel or PSA/TrustZone-type technology.

2.5 Cryptographic functionality

These are common cryptographic functions that a platform can provide that are useful to a product developer, but typically not directly 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 must verify all aspects of the parts of standards referenced.

2.5.1 Cryptographic operation

The platform provides the application with <cryptographic operations such as encryption, decryption, hashing, signing, signature verification> functionality as specified in <ISO/RFC/FIPS/RSA/NIST/SOGIS/OSCCA/… spec section x.y> for keylengths <> and modes <>.

Value

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

Considerations

The cryptographic operation must keep the confidentiality of the secret keys against the attacker. With the standard attacker model, timing and padding oracle attacks must be considered.

2.5.2 Cryptographic key generation

The platform provides the application with a way to generate cryptographic keys for use in <cryptographic algorithms> as specified in < ISO/RFC/FIPS/RSA/NIST/SOGIS/OSCCA/… spec section x.y> for keylengths <range>.

Value

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

Considerations

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.

2.5.3 Cryptographic keystore

The platform provides the application with a way to store <secret data, such as cryptographic keys and passwords> such that not even the application can disclose 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 can not be disclosed accidentally, provided that the keys are stored only in the keystore.

Considerations

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.

2.5.4 Cryptographic random number generation

The platform provides the application with a way based on <physical noise, cryptographic computation, combined physical noise and cryptographic computation> to generate random numbers to as specified in < ISO/RFC/FIPS/RSA/NIST/SOGIS/OSCCA/… spec section x.y>.

Value

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

Considerations

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 must always to be checked against.

2.6 Compliance functionality

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

These features are typically not easily visible to either the application developer or the product user. Features that are typically visible, such as "Secure communication", are listed elsewhere.

2.6.1 Secure encrypted storage

The platform ensures that all data stored by the application, with the exception of <data that is in the plain, for example for output to memory cards>, is encrypted as specified in <ISO/RFC/FIPS/RSA/NIST/SOGIS/OSCCA/… spec section x.y> with a platform instance unique key of keylengths <>

Value

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

Considerations

The key used to encrypt the data must 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.

Note that if the storage is not encrypted but otherwise protected against read out by physical attackers, "Physical attacker resistance" should be used.

2.6.2 Residual information purging

The platform ensures that <data>, with the exception of <data that is not erased automatically>, is erased using the method specified in <ISO/RFC/FIPS/RSA/NIST/SOGIS/OSCCA/… spec section x.y> 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

Typically <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 coldboot need to be considered.

2.6.3 Audit log generation and storage

The platform generates and maintains an audit log of <all significant security events> and allows access and analysis of these logs to authorized users <list of users> only.

Value

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

Considerations

"Software attacker resistance: isolation of platform" might also be claimed.

2.6.4 Reliable time

The platform provides the application with a measure of time with an accuracy of <resolution and maximum drift>.

Value

Evaluators and developers of composites can be ensured that there is a time source, without further activities.

Considerations

-

3 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, SESIP1+, SESIP2, SESIP2+ and SESIP3** and are hierarchical:

SESIP Assurance Level 1 (SESIP1) is a self-assessment-based level: the developer has to provide a simplified Security Target, describing the security claims of his product, together with a rationale why he believes these claims are met. Only minimal evaluator effort is needed: checks on consistency and clarity of the self-assessments are performed. There is no independent check by the evaluators the platform actually implements the SFRs.

SESIP Assurance Level 1+ (SESIP1+) 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. SESIP1+ provides a *moderate level* of assurance.

SESIP Assurance Level 2 (SESIP2) 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. SESIP2 provides a *substantial level* of assurance.

SESIP Assurance Level 2+ (SESIP2+) is currently a placeholder.

SESIP Assurance Level 3 (SESIP3) is the same as the traditional full CC evaluation against an EAL4+ALC_DVS.2+AVA_VAN.5 level that is used for smartcards, secure elements, e-passports etc. SESIP3 provides a *high level* of assurance. During the trial this level is exclusively for re-use of SOG-IS certified platforms by licensed labs, allowing those platforms to utilize the mappings from SESIP to specific commercial product domains immediately.

New assurance requirements (ASE_REQ.3, ADV_IMP.3) are marked in **bold** in the overview tables. Assurance requirements with a significant refinement are marked in *italic* in the overview tables. Vulnerability analysis (AVA) shall use the rating tables described in "Attack potential rating".

Note that earlier [SESIP] versions used the term "IoT Platform Assurance Level x" or "ITPx" for the same levels. For clarity of communication, only these names have been changed to "SESIP Assurance Level x" or "SESIPx". There have been no changes to the content of these assurance levels. Older mentions of "IITPx" shall be read as "SESIPx".

3.1 SESIP Assurance Level 1 (SESIP1)

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

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

- A simplified Security Target with self-assessment rationale. ST templates are available on the SESIP website.

3.1.1 Objectives

SESIP1 provides assurance by using a simplified Security Target, augmented by a self-assessment rationale. The self-assessment is considered to be part of the Security Target, even if it is delivered separately (for example in the form of a completed questionnaire).

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

3.1.2 Assurance components

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*
ALC: Life-cycle support	ALC_FLR.2 Flaw reporting procedures

3.1.3 Security Target Requirements

ASE_INT.1 ST Introduction

As per CC part 3.

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 guidance documents.

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:

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 "Identification of platform type" must be included. The SFR "Secure update of platform" must be included or under the ALC_FLR.2 it must be argued why updates are not applicable.

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

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** ciphersuite
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 must 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 must 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 statement of security assurance requirements matches one of the SESIP assurance levels defined in chapter 3 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.

ASE_TSS.1 TOE summary specification

As per CC Part 3

The dependency on ASE_REQ.1 is considered to be fulfilled by ASE_REQ.3.

Refinement[2]:

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 must 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 must indicate what procedures cover ALC_FLR.2 if "Secure update of platform" is included in the Security Target.

3.1.4 Life-cycle Support Requirements

ALC_FLR.2 Flaw Reporting Procedures

As per CC Part 3.

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 must 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 SESIP1+ level, flaw reporting procedure steps not visible to the users may be omitted from the description as these are not verifiable by the evaluators.

[2] This was explicitly stated as the scope of TSS (ASE_TSS.1.5C) in the CC v2.3, unfortunately lost in transition to CC v3.1.

3.2 SESIP Assurance Level 1+ (SESIP1+)

SESIP1+ is a moderate level of assurance for (parts of) IoT platforms.

SESIP1+ 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 SESIP2.

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

- A simplified Security Target with self-assessment rationale. ST templates are available on the SESIP website.
- 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.

3.2.1 Objectives

SESIP1+ 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 behaviour.

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.

SESIP1+ 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).

3.2.2 Assurance components

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.1 Vulnerability survey

3.2.3 Security Target Requirements

As per section 3.1.3.

3.2.4 Development Requirements

ADV_FSP.4 Complete functional specification

Dependencies: ADV_TDS.1 Basic design

This dependency does not need to be fulfilled, as ASE_TSS.1 fulfils this sufficiently for the black-box application.

For the rest as per CC Part 3.

3.2.5 Guidance Documents Requirements

AGD_OPE.1 Operational User Guidance

As per CC Part 3.

AGD_PRE.1 Preparative Procedures

As per CC Part 3.

3.2.6 Life-cycle Support Requirements

ALC_FLR.2 Flaw Reporting Procedures

As per section 3.1.4.

3.2.7 Tests Requirements

ATE_IND.1 Independent testing: conformance

As per CC Part 3.

3.2.8 Vulnerability Analysis Requirements

AVA_VAN.1 Vulnerability survey

Dependencies:

AGD_OPE.1 and AGD_PRE.1 are met.

ADV_FSP.1 is met by ADV_FSP.4.

Note

The effort spent on AVA_VAN.1 (vulnerability analysis and penetration testing) is expected to be equivalent to 20 man day for a typical platform to reach Basic attack potential.

Definition of the "equivalent effort" and "typical platform" is currently at the discretion of the CB during the trial projects. See "Further development of SESIP".

3.3 SESIP Assurance Level 2 (SESIP2)

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

It provides significantly more assurance than SESIP1+ 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 high assurance provided by SESIP3.

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

- A simplified Security Target with self-assessment rationale. ST templates are available on the SESIP website.
- 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.

3.3.1 Objectives

SESIP2 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 behaviour.

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.

SESIP2 also provides assurance through the use of development environment controls and additional TOE configuration management including automation, and evidence of secure delivery procedures.

3.3.2 Assurance components

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.2 Vulnerability analysis

3.3.3 Security Target Requirements

As per section 3.1.3.

3.3.4 Development Requirements

ADV_FSP.4 Complete functional specification

Dependencies: ADV_TDS.1 Basic design

This dependency does not need to be fulfilled, as ADV_IMP.3.3D covers this.

For the rest as per CC Part 3.

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

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*".*

3.3.5 Guidance Documents Requirements

AGD_OPE.1 Operational User Guidance

As per CC Part 3.

AGD_PRE.1 Preparative Procedures

As per CC Part 3.

3.3.6 Lifecycle 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 that at SESIP2 level and higher, the entire flaw reporting procedure is to be described; not just the externally visible steps required for SESIP1 and SESIP1+ levels.

3.3.7 Tests Requirements

ATE_IND.1 Independent testing: conformance

As per CC Part 3.

3.3.8 Vulnerability Analysis Requirements

AVA_VAN.2 Focused vulnerability analysis

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.1 are considered to be met by ADV_IMP.3 in combination with ASE_TSS.1.

Note

The effort spent on ADV_IMP.3 is expected to be equivalent of at least 5 man days for a typical platform.

The effort spent on AVA_VAN.2 (vulnerability analysis and penetration testing) is expected to be equivalent to 25 man day for a typical platform to reach Enhanced-Basic attack potential.

Definition of the "equivalent effort" and "typical platform" is currently at the discretion of the CB during the trial projects. See "Further development of SESIP".

3.4 SESIP Assurance Level 2+ (SESIP2+)

SESIP2+ is currently a placeholder.

3.5 SESIP Assurance Level 3 (SESIP3)

SESIP3 is the same as the standard high assurance level currently being used for smartcards, secure elements, E-passports etc[3][4]. It provides a very robust defence against very advanced threats.

During the trial this assurance level is exclusively for re-use of SOG-IS certified platforms by licensed labs, allowing those platforms to utilize the mappings from SESIP to specific commercial product domains immediately.

3.5.1 Objectives

SESIP3 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 behaviour.

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.

SESIP3 also provides assurance through the use of 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.

3.5.2 Assurance components

Assurance Class	**Assurance Families**
ASE: Security Target evaluation	ASE_INT.1 ST Introduction ASE_CCL.1 Conformance claims ASE_ECD.1 Extended components definition ASE_OBJ.2 Security objectives **ASE_REQ.3 Listed Security requirements** ASE_SPD.1 Security problem definition *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

[3] See for instance the Security IC Platform Protection Profile BSI-PP-0084-2014.
[4] With the exception of the addition of ALC_FLR.2, as updating of flaws is considered a very important functionality for IoT platforms.

	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_LCD.1 Developer defined life-cycle model ALC_TAT.1 Well-defined development tools
ATE: Tests	ATE_COV.2 Analysis of coverage ATE_DPT.1 Testing: basic design ATE_FUN.1 Functional testing ATE_IND.2 Independent testing - sample
AVA: Vulnerability Assessment	AVA_VAN.5 Advanced methodical vulnerability analysis

All but ASE_REQ.3 are as per CC part 3. Optimized methodology is available.

3.6 Attack potential rating

The attack potential shall be calculated with the following rating tables, replacing the general rating table of the CC.

Note that these rating tables are consistent with the rating used in smart card devices as described in [AM].

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. SESIP3).
Restricted	2	2	
Sensitive	4	3	
Critical	6	5	
Very critical hardware design	9	NA	
Access to TOE			
< 10 samples	0	0	It is unlikely that evaluations under SESIP3 will rate more than 10 samples in the attack.
< 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 (besides the evaluation lab) are included in the scope of the evaluation at ALC_DVS.2 level (i.e. SESIP3).
Restricted	2	NA	
Sensitive	4	NA	
Critical	6	NA	

The table below shows what the total rating of identification + exploitation needs to be to meet the attack potentials.

SESIP				Standard CC rating (in [CEM]) for comparison[5]		
	Rating of at least	(Range of values)	**Meets attack potential of**	Rating of at least	(Range of values)[6]	Meets attack potential of
SESIP Assurance Level 1 (SESIP1)	**-**[7]	**-**	**-**	-	-	-
SESIP Assurance Level 1+ (SESIP1+)	**16**	0-15	**Basic (AVA_VAN.1 and AVA_VAN.2)**	14	14-19	Enhanced basic (AVA_VAN.3)
SESIP Assurance Level 2 (SESIP2)						
	21	16-20	**Enhanced-Basic (AVA_VAN.3)**	20	20-24	Moderate (AVA_VAN.4)
	25	25-30	**Moderate (AVA_VAN.4)**	25	≥25	High (AVA_VAN.5)
SESIP Assurance Level 3 (SESIP3)	**31**	31 and above	**High (AVA_VAN.5)**	Does not exist		

[5] This shows that the rating of SESIP and [AM] at AVA_VAN.x is effectively exceeding the standard CC AVA_VAN.x+1 level, due to the higher minimal ratings required in

[6] The failing values are indicated in ranges in the [AM] and [CEM] documents, the minimal passing value is listed in the "Rating of at least" column.

[7] There is **no** vulnerability analysis or penetration testing at this level, only self-assessment.

4 Governance

TrustCB BV governs the Security Evaluation Scheme for IoT Platforms (SESIP) scheme as the scheme owner. It is also (currently) the only Certification Body (CB).

The procedures for platform certifications and lab licensing can be found on the scheme website <https://trustcb.com/iot/>.

4.1 Workgroups

The SESIP scheme is set up in an open manner in accordance to the WTO Technical Barriers to Trade and general best practices in the security evaluation and certification domain.

The following workgroups are applicable for SESIP:

- SESIP user group: General development and promotion of SESIP.
- JHAS: development and alignment of attack ratings, especially for use in SESIP Assurance Level 3 (SESIP3).
- TBD: development and alignment of attack ratings, especially for use in SESIP Assurance Level 1+ (SESIP1+) and SESIP Assurance Level 2 (SESIP2).

4.2 Derived dedicated schemes and profiles

It is possible and intended to derive dedicated schemes from SESIP focussing on a specific domain (such as a business vertical, specification, or technology). The objective of such derived dedicated scheme would be to:

- Standardising and aligning (minimal) security functional requirements and SESIP assurance level for that domain, for instance by developing profiles.
- Further optimize the evaluation activities, for instance by defining a more precise set of functional tests to be performed.
- Further reduce the development effort.

Derived schemes will need to follow the SESIP governance rules. Their management may associate the owners (company, standard body, trade association, …) of the technology behind the derived scheme, significant stake holders (developers, evaluators, certifiers, users, …), or their representatives. The details of this management will be defined on a case-by-case basis for the first derived schemes.

5 Example use cases

5.1 Examples for specific use cases

This scheme provides 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.

5.1.1 Secure update of a device (OTA)

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

- Identification of platform type:
  The platform provides a unique identification of the platform type, including all its parts and their versions.
- Secure update of platform
  The platform can be updated in the field such that the integrity, authenticity and confidentiality of the platform is maintained.
- Secure update of application:
  The product can be updated in the field such that the integrity, authenticity and confidentiality of the product is maintained.

A more extensive platform supporting checks that the product is genuine and operating correctly, would add:

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

5.1.2 A blood glucose measurement device (DTSec)

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

This product will need to fulfill DTSec requirements.

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

- Identification of platform type:
  The platform provides a unique identification of the platform type, including all its parts and their versions.
- 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 and modification** of messages between these endpoints, using **Bluetooth LE 4.1 with options x,y,z always enabled**.

- Cryptographic operation:
  The platform provides the product with **hashing, signing, and signature verification** functionality as specified in **<DTSec compliant references here>** for keylengths of **1024-2048 bit** and modes **PKCS#11**.
- 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>**.

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 the "Genuine application" and "Attested state of application" return the right values, and that the signature setting and verification is implemented in the way intended:

- Identification of platform type:
  The platform provides a unique identification of the platform. This is uniquely identifies all components of the platform (hardware and software components) and their versions.
- Genuine application:
  The platform provides an attestation of the application, in a way that cannot be cloned or changed without detection.
- Attested state of application:
  The platform provides an attestation of state of the application.
- Secure update of platform:
  The platform can be updated in the field such that the integrity, authenticity and confidentiality of the platform is maintained.
- 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.
- Secure communication support:
  The secure communication channel authenticates **any external party** and protects against **disclosure and modification** of messages between the endpoints, using **Bluetooth LE 4.1 with options x,y,z always enabled**.
- 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.
- 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 keylength of **128 bits.**

The product will still need to implement (and be evaluated) to properly sign/verify data exchanged, so the platform needs to provide:

- Cryptographic operation:
  The platform provides the product with **hashing, signing, and signature verification** functionality as specified in **<DTSEC compliant references here>** for keylengths of **1024-2048 bit** and modes **PKCS#11**.
- 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>**.

6 Further development of SESIP

SESIP will grow and adapt with the experience gained in the application during evaluations and certifications. And to actually get going with gathering the experience, we have to decide not to wait until it is in the mythical perfect state.

Below are the topics that we already know will be addressed in the further development of SESIP:

- More explicit description of the assets and their protection. The confidentiality of the user data is clearly an asset to be protected (but the platform doesn't really know what is user data, the application knows). Confidentiality of the application code is maintained: it is unclear whether the application code was originally considered confidential, but at least the update functionality don't break the confidentiality. Etc.
- SFRs for other standard functionality. Currently identified as likely useful to have in a platform:
    - "Secure debug": authentication is needed to activate debugging (simple version already available in "Decommission of platform")
    - "User authentication": PIN/password/biometrics/… handling (simple version already available in "Cryptographic keystore")
    - "Access control to security functionality/keys/credentials" (simple version already available in "Cryptographic keystore" and "Software attacker resistance: isolation of platform parts")
    - "Trusted provisioning": functionality for secure production in less-secure environments (simple version already available with "Secure communication support" and "Secure install of application"), or assurance for more-secure environments.
    - Standard OS functionality like external input validation, closing of ports, firewalling, limitations of connections.
- (Protection) Profiles capturing standard requirements of specific domains/verticals, and profiles capturing standard functionality delivered by a certain type of platform (such as ARM PSA, Eurosmart PP-0084, Oracle Java Card, IEC 62443, …). Drafting has already started.
- Clear description of the additive composition approach, including how to formulate multi-assurance TOEs.
- Methodology for SESIP1+ and SESIP2 attack rating. This may impact the "20/25 man day equivalent testing" of SESIP1+/SESIP2.
- Methodology for efficient SESIP1 evaluation.
- Explicit description of the life-cycle stages considered, including end of manufacturing and end of use by this user (Factory reset of platform) or in general (Decommission of platform), life cycle transitions.

See "Workgroups" or contact the scheme owner TrustCB <SESIP@trustcb.com> for inquiries how to be part of this process.

7 References

7.1 Terminology

CC	Common Criteria
CEM	CC Evaluation Methodology
ICD	In-Circuit Debugger, allows for debugging of a live platform.
JHAS	JIL Hardware Attacks Subgroup, workgroup under Eurosmart and JIL, maintaining and further developing the industry-standard attack rating for integrated ICs and similar devices.
JTAG	Standardised interface for debugging, testing and programming devices.
SmartCC	Project code name for the Smart application of the CC, such as the MIFARE v3.0 scheme and the SESIP scheme

7.2 Bibliography

[AM]	Application of Attack Potential to Smartcards, Joint Interpretation Library, version 2.9, dated January 2013
[CC]	Common Criteria for Information Technology Security Evaluation, Parts I, II and III, Version 3.1 Revision 5, April 2017
[CEM]	Common Methodology for Information Technology Security Evaluation, Version 3.1 Revision 5, April 2017