Global Platform®

The standard for secure digital services and devices

GlobalPlatform Technology

SESIP Profile for Code Update Mechanism

Version 0.0.0.12

Public Review

August 2024

Document Reference: GPS_SPE_026

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

Tables

1 INTRODUCTION

Keeping software updated with the latest security patches and bug fixes is widely agreed (see [EN 17927], [SESIP], [UNR 156]) as necessary for any device that claims security, safety, or privacy protection.

In this document, we use the term *Code Update Mechanism* to refer to the variety of technical software and hardware means implemented in the device and its operational environment, as well as the management actions required to operate them to perform software updates and code patching.

This document defines requirements, technical guidelines, and recommendations for implementing the Code Update Mechanism securely and efficiently:

- Requirements are marked as REQ,
- Recommendations are marked as REC.

Conformance Claims

This SESIP Profile claims conformance to CC Part 2 and SESIP Assurance Level 2. In addition, section 7 of this document addresses the requirements for SESIP Assurance Level 3.

1.1 Audience

This document is intended primarily for security architects and platform and system developers.

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 has 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
SESIP Methodology	GlobalPlatform Technology Security Evaluation Standard for IoT Platforms (SESIP) Methodology v1.2, July 2023	[SESIP]

Table 1-2: Informative References

Standard / Specification	Description	Ref
EN 17927	Security Evaluation Standard for IoT Platforms, November 2023	[EN 17927]
UN Regulation No. 156	Software update and software update management system, Rev 3, March 2021	[UNR 156]

1.4 Terminology and Definitions

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

Table 1-3: Terminology and Definitions

Term	Definition
SESIP Profile	A security profile generic to a type of platform (part). Equivalent to CC Protection Profile: A generic SESIP Security Target defining the SESIP requirements in terms of security features and evaluation activities to be addressed during the evaluation of a platform (part) of the type targeted by the profile.

1.5 Abbreviations

Table 1-4: Abbreviations

Abbreviation	Meaning
CC	Common Criteria
SESIP	Security Evaluation Standard for IoT Platforms
SFR	Security Functional Requirement

1.6 Revision History

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

Table 1-5: Revision History

Date	Version	Description
May 2024	0.0.0.10	Committee Review
June 2024	0.0.0.11	Member Review
August 2024	0.0.0.12	Public Review
TBD	v1.0	Public Release

2 SOFTWARE UPDATE IN LIFE CYCLE

The Security Evaluation Standard for IoT Platforms ([SESIP]) outlines various life cycle models and emphasizes security during significant phases before and after the normal usage phase.

However, within this normal usage phase, connected devices experience a continuous cycle of software updates, and the life cycle analysis should address this. Indeed, promptly developing and deploying security updates is critical for companies to safeguard their customers and contribute to the overall health of the technical ecosystem.

2.1 Scope of Software Updates

REC All software components within connected IoT devices should be securely updateable.

Vulnerabilities often arise from non-security-related software components. Therefore, it is essential to keep all software up-to-date and well-maintained.

2.2 Software Update Availability

REQ When a code update is necessary, it must be applied promptly.

To achieve this:

- Device users should be informed about the need for the update.
- If user action is required, they should be prompted to perform the update.
- However, executing software updates without user involvement is preferable whenever possible.

2.3 Software Update Feasibility

REC A code update should be easy to implement.

Preferably, it should occur in the background without disrupting the device's functionality until the updated version is activated.

3 SECURITY PROBLEM DEFINITION

3.1 Assets

Table 3-1: Assets

Asset	Description	Protected Attributes
Code Update Image	The code downloaded to the Platform to update or replace the current TSF or application code in whole or partially	Integrity, authenticity, freshness Optionally – confidentiality
Code Update Version	The attribute of the Code Update Image presented to the Platform during the code update process	Authenticity
Current Code Version	The version of the last successfully installed Code Update Image stored persistently in the Platform	Authenticity, rollback protection
Code Update Authentication Key	The public key or secret key used to verify the authenticity of a presented Code Update Image	Integrity For a secret key, also confidentiality If updatable – authenticity
Code Update Confidentiality Key	The private key or secret key used to decrypt the Code Update Image	Confidentiality If updatable – authenticity

3.2 Users / Subjects

- Code Update Deployer
- Platform User

3.3 Threats

- Blocking Code Updates
- Forging Update Deployer Authorization
- Code Update Mechanism Abuse
- Rollback of a Code Update
- Partial Code Update
- Code Update Image Disclosure

3.4 Organizational Security Policies

REQ	If a Code Update Authentication Key is a secret key (i.e., not a public part of an asymmetric key), it shall be individual for each instance of the Platform.
REQ	Code Update Authentication and Confidentiality Keys shall be generated with the required cryptographic strength and entropy amount. This policy belongs to the general Key Generation category and is outside the scope of this document.
REQ	Code Update Authentication and Confidentiality Keys shall be securely provisioned to the Platform and kept secret by the Platform and at the Update Deployer's facility. This policy belongs to the general Key Management category and is outside the scope of this document.

3.5 Assumptions

- The Code Update Deployer uses the correct authentication and confidentiality keys for Code Update Image protection.
- The Code Update Issuer increments the Code Update Version for subsequent code updates.

4 SECURITY OBJECTIVES

4.1 Current Code Version Attestation

The Code Update Deployer shall be able to obtain the Current Code Version of the Platform to decide whether the code running on the Platform is outdated.

4.2 Code Update Availability

The Platform should employ countermeasures to prevent an attacker from blocking code updates for the Platform.

At least, the Code Update Deployer shall inform the Platform User when a code update is scheduled for the Platform.

4.3 Code Update Feasibility

The Code Update process shall not risk the functioning of the Platform during and after the update.

4.4 Code Update Authenticity and Integrity

The Platform shall reject non-authentic or modified Code Update Images.

4.5 Rollback Protection

The Platform shall protect from applying outdated code updates, e.g., by ensuring that the code version increases on update.

4.6 Atomicity of the Code Update

The Platform shall conduct the code updates in an all-or-nothing manner; i.e., prohibiting the execution of any part of the new code before and any part of the deprecated code after the successful activation of the new code.

4.7 Secure Fail

If a failure occurs during Code Update activation, the Platform shall retreat to a secure state without compromising its security assets and, preferably, without losing its functionality.

4.8 Patch Confidentiality

The Platform may optionally provide a secure communication channel for Code Update Image retrieval, allowing its contents to be encrypted before activation.

5 SECURITY FUNCTIONAL REQUIREMENTS

This section iterates through the relevant SESIP SFRs ([SESIP] section 3), analyzing their application to cover the Security Objectives (SO) introduced in section 4.

5.1 Verification of Platform Instance Identity

Requirement (from [SESIP]): The Platform provides a unique identification of that specific instantiation of the Platform, including all its parts.

REQ	The Platform shall provide means for verifying the current version of the platform code and the applications.

The Code Update Deployer needs this verification to determine whether the code has been updated.

5.2 Attestation of Platform Genuineness

Requirement (from [SESIP]): The Platform provides an attestation of the "Verification of Platform Identity" and "Verification of Platform Instance Identity", in a way that ensures that the Platform cannot be cloned or changed without detection.

REC	The Platform should provide a cryptographic mechanism to prevent an attacker from interfering with verifying the current version of the platform code.

This mechanism prevents an attacker from falsifying the Current Code Versions of the Platform code.

5.3 Attestation of Application Genuineness

REC	The Platform should provide a cryptographic mechanism to prevent an attacker from interfering with verifying the current version of the Application code.

Like section 5.2, but applies to the Application code update.

5.4 Secure Initialization of Platform

Requirement (from [SESIP]): The Platform ensures its integrity and authenticity during Platform initialization. If the platform integrity or authenticity cannot be ensured, the Platform will go to <list of controlled states>.

REQ	Before the updated code is executed the first time, the Platform must verify that the code update has been executed in its entirety. If not, either the previous version of the code shall be restored, or the Platform shall retreat to a secure state without compromising its security assets.

Once the code update is complete, applying the new code requires restarting the Platform or the Application. Following the restart, the Platform shall enforce this SFR.

5.5 Secure Update of Platform

Requirement (from [SESIP]): The Platform can be updated to a newer version in the field such that the <confidentiality,> integrity and authenticity of the Platform are maintained.

Properly implementing this Security Functional Requirement (SFR) is the central focus of this entire document. At a minimum, the Platform must incorporate a mechanism that ensures the following SFRs:

5.5.1 Code Update Availability

REC Watchdog timers, expirable entitlements, or similar mechanisms should be employed to prevent an attacker from obstructing code updates for the Platform.

5.5.2 Code Update Feasibility

REQ The device must maintain essential functionality, critical for remaining available during a Platform code update.

5.5.3 Performing Image Download in Background

REC If the code update image download takes time, performing it in the background without disrupting the device's overall functionality is advisable. Subsequently, the new code can be activated using memory remapping or a similar mechanism.

5.5.4 Code Update Authenticity and Integrity

REQ The authenticity of the presented Platform code update image shall be verified using a dedicated Code Update Authentication Key.

5.5.5 Asymmetric Signature Mechanism

REC Preferably, an asymmetric signature mechanism should be used for Platform code update verification, which does not necessitate the confidentiality of the image verification key.

5.5.6 Rollback Protection

REQ The Platform must prevent updates where the Platform Code Update Version is smaller than the Current Platform Code Version.

Note: By implementing this restriction, the Current Code Version becomes a Reliable Index as defined in [SESIP] section 3.6.7.

5.6 Secure Update of Application

5.6.1 Code Update Availability

REC	Watchdog timers, expirable entitlements, or similar mechanisms should be employed to prevent an attacker from obstructing code updates for the Application.

5.6.2 Code Update Feasibility

REQ	The device must maintain essential functionality available during an Application code update.

5.6.3 Code Update Authenticity and Integrity

REQ	The authenticity of the presented Application code update image shall be verified using a dedicated Code Update Authentication Key.

5.6.4 Rollback Protection

REQ	The Platform must prevent updates where the Application Code Update Version is smaller than the Current Application Code Version.

The same as section 5.5, but applies to the Application code update.

5.7 Secure Communication Support

Requirement (from [SESIP]): The Platform provides 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>.

REQ	The Platform shall ensure that the code updates originate from an authorized Code Update Deployer and are delivered over a secure channel.

6 RATIONALE FOR SECURITY FUNCTIONAL REQUIREMENTS

Table 6-1: Rationale for Security Functional Requirements

Security Objective	Security Functional Requirements
4.1 Current Code Version Attestation	5.1, 5.2, and 5.3
4.2 Code Update Availability	5.5.1 and 5.6.1
4.3 Code Update Feasibility	5.5.2, 5.5.3, and 5.6.2
4.4 Code Update Authenticity and Integrity	5.5.4, 5.5.5, and 5.6.3
4.5 Rollback Protection	5.5.6 and 5.6.4
4.6 Atomicity of the Code Update	5.4
4.7 Secure Fail	5.4
4.8 Patch Confidentiality	5.7

7 SECURITY ASSURANCE REQUIREMENTS

REQ	Secure Consumer IoT devices certified to the SESIP Assurance Level 2 shall implement the Code Update mechanism according to the Security Functional Requirements listed in section 5.
REQ	For Secure Consumer IoT devices certified to the SESIP Assurance Level 3 and higher, the Recommendations listed in section 5 shall be considered as Requirements.
REQ	For constrained devices that do not implement a Code Update mechanism, the Platform developer shall provide a transparent rationale for the absence of software updates.