Security Target for STM32MP13xx

Document information

This security target document is based on GlobalPlatform® Security Evaluation Standard for IoT Platforms (SESIP), version 1.1 (June 2021), GP_FST_070.

1 Introduction

This Security Target describes the STM32MP13xx platform and the exact security properties of the platform that are evaluated against GlobalPlatform® Security Evaluation Standard for IoT Platforms (SESIP) [1].

This security target is inspired by GlobalPlatform® Protection Profile referenced in [5], but it is not compliant to it.

This Security Target covers critical security functional requirements for the development of PCI PTS POI compliant devices based on the STM32MP13xx Platform, according to [SR] and [DTR]. The expectation is the platform can be labelled as “PCI ready” by the Security Evaluation Laboratory.

Table 1. Protection Profile Reference and Conformance Claims

Reference	Value
Protection profile name and version	
Assurance claim	See Section 3.1

1.1 Security Target Reference

This document: Security Target for STM32MP13xx, Revision 5c (January 2023), STMicroelectronics. DocID: cx801422.

1.2 Platform Reference

Table 2. Platform reference

Reference	Value
Platform name	STM32MP13xx
Platform version	1.2 (RevID= 0x1003)
Platform identification	STM32MP13xC, STM32MP13xF
Platform type	Microprocessor platform with its embedded ROM, for IoT, industrial, or consumer applications.

1.3 Included Guidance Documents

The following documents are included with the platform:

Table 3. Guidance documents

Reference	Name	Version
UM2885 [2]	STM32MP13xx Security Guidance	2
RM0475 [3]	Reference manual, STM32MP13xx advanced Arm®-based 32-bit MPUs	0.5
UM2237 [4]	User manual, STM32CubeProgrammer software description	18

1.4 Platform Functional Overview and Description

The STM32MP13xx microprocessor is the SESIP certified member of the STM32MP13 family of general-purpose microprocessor solution (MPU). It provides a new optimal balance between performance, power and security.

The platform consists of an ARM Cortex-A7 based microprocessor with an immutable ROM firmware, and with key security hardware features like TrustZone, secure embedded SRAM, antifuses, and crypto accelerators.

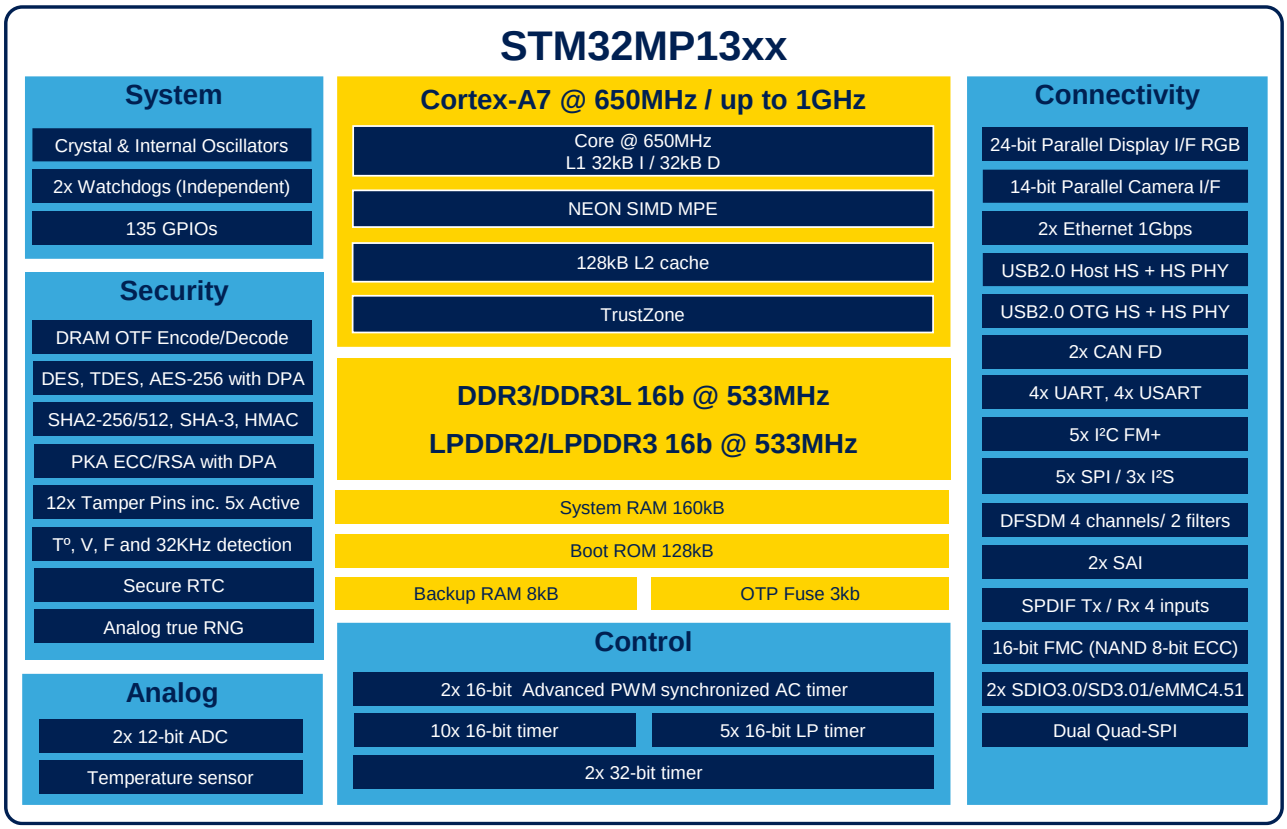
1.4.1 Platform Security Features and Scope

The STM32MP13xx microprocessor is designed with a comprehensive set of security features, some of them based on the standard Arm TrustZone® technology. Those features include:

- A 32-bit Cortex-A7 core with TrustZone, 32-kB L1 Data / 32-kB L1 Instruction cache + 128kB unified L2 cache
- A set of hardware crypto blocks with symmetric crypto functions (SAES and CRYP), asymmetric crypto accelerator (PKA) and hash function (HASH)
- A tamper detection block (TAMP) with up to 12 tamper pins
- 3072 fuse bits, used for unique ID storage & 256 bits Hardware Unique key (BSEC and RHUK)
- A set of SRAM memories with security & anti-tamper measures (SRAM3, Backup RAM and PKA RAM)

An overview of the STM32MP13xx microprocessor is shown in the block diagram below.

Figure 1. STM32MP13xx block diagram



Secure boot ROM in the integrated circuit includes the following features:

- Secure boot
  - FSBL loading from the boot device (flash memory or serial link)
  - FSBL authentication and optional decryption
  - FSBL launching
- Wake-up from low power mode
- Secure Secret Provisioning of the OEM sensitive assets in antifuses
- Silicon device life-cycle support (return material for analysis)

TOE Type

The Target of Evaluation (TOE) consists of the following components:

- The STM32MP13xx microprocessor device
- The boot ROM embedded in it

TOE Physical Scope

The physical scope of the TOE is the STM32MP13xx integrated circuit, identified as defined in Section 1.2. The hardware interfaces of the TOE are listed in Section 4.2.2 of [2].

TOE Software Scope

The logical scope of the TOE is defined in Table 4. Any additional firmware, OS or application software stored on the platform is not in scope of this evaluation.

Table 4. Software Components and Interfaces of the TOE

Component/Interface	Description	Identification/Version
Boot ROM	Device embedded ROM code (same version as the silicon)	1.2

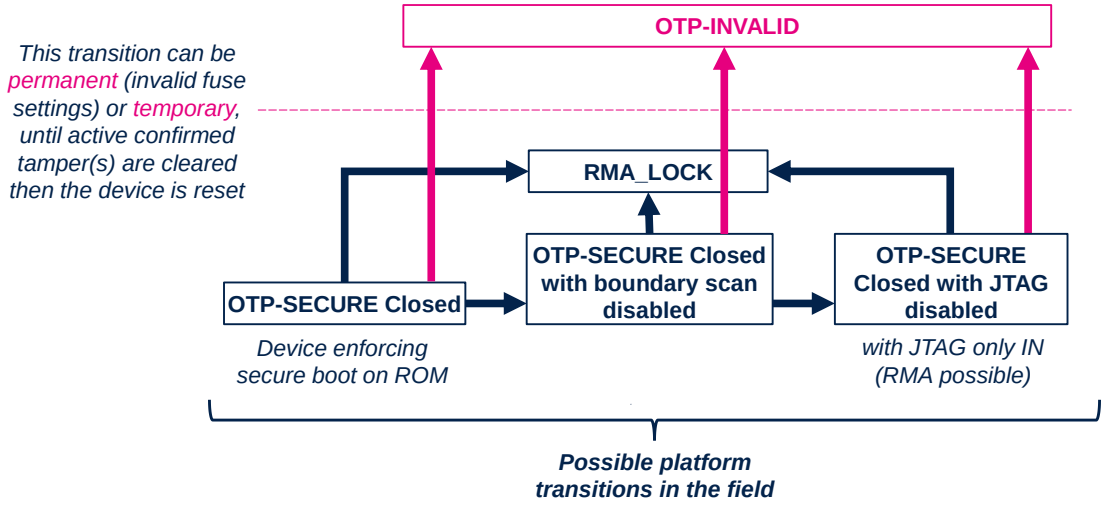
No additional non-platform hardware, software or firmware is required for the correct functioning of the security claims described in this document.

1.4.2 Life Cycle

Transitions to the lifecycle states of the platform are irrevocably triggered by burning fuse bits in BSEC. Those transitions are summarized on the figure below.

In OTP-SECURE Closed states, the TOE is certified with all debug features disabled out of reset. This debug protection can be frozen anytime until next reset by setting the DENREG bit in BSEC peripheral.

Figure 2. Platform lifecycle overview



The transition to RMA_LOCK state is initiated by the integrator using its own 32-bit password. After three consecutive wrong password attempts, the RMA sequence always fails.

The integrator must store its security-sensitive information (secrets keys, certificate, RMA key) in OTP words 32 to 95. It is mandatory in order not to leak secret information when device transits to RMA_LOCK state.

1.4.3 Use Case

The TOE is intended to be used by a trusted integrator as a SESIP Level 3 compliant Root-of-Trust platform. Integrator would add on top of it the required components to make a connected product. Such components include a Root of Trust software layer, an operating system, some connectivity, as well as additional hardware components as required by the final product.

2 Security Objectives for the Operational Environment

2.1 Platform Objectives for the Operational Environment

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

- TOE_SECRETS: protection of root of trust related sensitive material, as described in section 4.2.4 of [2].
- TOE_PREPARATION: after verifying TOE genuineness (section 3.1 of [2]) the integrator personalizes the TOE following the TOE documentation in section 4.2.4 of [2].
- TRUSTED_INTEGRATOR: the integrator uses the security functionalities of the TOE in certified configuration following the TOE documentation in section 4.2.4 of [2]. The integrator is trusted and does not attempt to thwart the TOE security functionalities nor attempt to bypass them.

2.2 Inherited Objectives for the Operational Environment

The platform does not include platform parts that have previously been evaluated under any SESIP certification scheme.

3 Security Requirements and Implementation

3.1 Security Assurance Requirements

The claimed assurance requirements package is **SESIP3**, as defined in Chapter 4 of GlobalPlatform® Technology Security Evaluation Standard for IoT Platforms (SESIP) [1].

3.2 Flaw Reporting Procedure (ALC_FLR.2)

The SFR “Secure update of platform” is not applicable, since the platform does not support the patching of the immutable ROM firmware. Customer can implement their own secure update mechanism in their code.

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:

Vulnerability reporting

To report a security vulnerability impacting a STM32 product or solution, you should contact PSIRT team through psirt@st.com as referred on the www.st.com/psirt web page.

To allow ST Product Security Incident Response Team (PSIRT) processing the potential discovered security vulnerability, you should provide the following information:

- ST product identification: part number or product reference and version (hardware or software)
- Complete technical description of the potential vulnerability, including any related known exploits
- How and when the potential vulnerability was discovered
- Any public information already published or publication planning (CVE, academic paper publication, etc.)
- Your contact information to use during the process

Due to the sensitivity of vulnerability information, it is recommended to provide your findings through encrypted email using the below ST PSIRT PGP/GPG key.

Vulnerability management process

Once submitted, ST PSIRT will manage the reported vulnerability according to the following process:

- 1. Reporting a new vulnerability: At this stage, ST PSIRT will acknowledge the reception of the reported issue.
- 2. Evaluating: ST PSIRT will evaluate the potential vulnerability to understand if there is an issue, analyze it, and set a priority to manage valid issues. ST PSIRT may come back to the submitter in case some information is missing from the original report or if clarification is needed.
- 3. Solving: ST PSIRT will investigate potential solutions and mitigations to address the issue.
- 4. Communicating: Once a solution is available (fix or mitigation), ST PSIRT will communicate back to the submitter and others where appropriate.

3.3 Security Functional Requirements

The platform fulfills the following security functional requirements:

3.3.1 Verification of Platform Identity

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

Conformance rationale:

The platform referred in Section 1.2 provides the following unique identifications:

- Integrated circuit hardware revision (RevID) and DieID, readable using the debugger or via USB

Verification methods and expected values are summarized in Section 3.1 of [2].

3.3.2 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 detailed in “Errors” description of “Flash memory interfaces” and “Serial boot interfaces” in section 4.2.2 of [2].

Informational:

This SFR contributes to meet the PCI PTS POI requirement B1 as it provides a proof of integrity and authenticity of the platform. In particular, it will be useful for the test cases TB1.1, TB1.10 and TB1.11 (see [DTR]).

Conformance rationale:

HW initialization steps

Following the chain of trust principle embedded ROM code in the device always manages system reset events, split into two categories:

- an application reset i.e. VDD power-on, reset button press, an HSE clock failure, an IWDG1/2 reset, a VDD brownout reset or the setting of MPSYSRST bit in the RCC.
- A wake-up from low power mode (VDDCORE power-on reset)

According to the execution condition, and fuse values, ROM code can:

1. Decide to proceed with the boot chain, attempting to execute genuine First Stage Boot Loader (FSBL) code from embedded SRAM.
2. Discover that a confirmed tamper event is blocking the access to fuses and has erased secrets in the device. In this case ROM code transforms this confirmed tamper as potential tamper, to execute the OEM FSBL following a system reset.
3. Detect fuse perturbations that prevent boot chain execution (e.g. invalid OTP security state). In this scenario only possible outcome is an application reset.
4. Branch to the LPLV-Stop2 exit firmware located in embedded SRAM.

Regarding platform integrity, it is enforced by hardware in all of the above cases:

- In case 1: ROM code uses immutable fuses provisioned by OEM to verify integrity of FSBL before executing it from embedded SRAM, robust against hardware fault injections (see next subsection).
- In case 2 and 3: Only trusted ROM code executes, leading to system reset. When the fuse errors are uncoverable case 3 leads to a state where the chip cannot be exploited as it will never execute any external code, keeping secrets locked from debugger and test modes.
- In case 4: Standby exit firmware has been verified in authenticity/integrity before going to Standby mode.

SW Initialization Steps

ROM code is in charge of loading, checking, optionally decrypting and launching the First Stage Boot Loader (FSBL) that is stored in an external Flash memory.

When the chip boots, the processing starts by the ROM execution which analyzes chip state in accordance with life cycle (TOE lifecycle is shortly described in Section 1.4.2).

As TOE is in OTP-SECURE Closed state, the ROM applies secure boot mechanisms. It verifies integrity and authenticity of the FSBL by verifying an elliptic curve digital signature. The algorithm used is a 256-bit ECDSA (NIST prime256v1 or brainpoolP256t1) signature of a SHA256 message digest of the FSBL code.

The public key used to sign the FSBL is verified against 8 possible root keys, which fingerprint is stored in the device’s manufacturer-programmable on-chip fuses.

The ROM verifies that the version of the FSBL is higher or equal than the current version installed, preventing roll-back and downgrading attacks.

The ROM decrypts the FSBL and verifies the plain text integrity when the decryption is required.

In case of failure, the ROM first look for a recovery boot FSBL (the flash memory may contain several copies of FSBL), before waiting on a Serial Downloader (on UART or USB) for an authenticated and optionally encrypted FSBL.

For more details, refer to section 4.2.1 of [2].

3.3.3 ~~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.~~

Non-conformance rationale:

The absence of this functionality is explained in Section 3.2 “Flaw Reporting Procedure (ALC_FLR.2)”.

3.3.4 Residual Information Purging

The platform ensures that all SRAM used by the platform, with the exception of SRAM not used by the platform, is erased using the method specified in this section before the memory is used by the platform or application again and before an attacker can access it.

Conformance rationale:

Following a reset ROM code erases SRAM contents before using the memory.

The boot ROM erases all SRAM area and registers which has contained secrets with random values after usage.

The boot ROM also erases all its SRAM data before jumping to the authenticated application.

3.3.5 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.

Informational:

This SFR contributes to meet the PCI requirements A1, A3, A4, A5, A6, A7, A8, and indirectly to A10 and A13 (see [DTR]).

Conformance rationale:

The platform provides the following countermeasures against physical attacks:

- ROM code execution hardening using redundancy checks and time jittering
- Tamper-detection & response hardware (i.e. automatic erase/blocking if confirmed tamper), defining in the device a protected area for following sensitive functions:
    - Non-volatile fuse storage
    - Volatile secret key storage in backup domain registers
    - Embedded SRAM3 storage
    - Crypto functions in hardware engines (SAES, CRYP, HASH, PKA)
- Detection of transient perturbation attacks in crypto functions (SAES, PKA private operations)
- External clock glitch filtering and clock loss detection:
    - For LSE clock, in all system modes, including VBAT mode
    - For HSE clock (input of the PLL feeding the Cortex A7) loss can be detected in Run mode and in Stop modes when the HSE oscillator is enabled. PLL filters glitches.
- Detection of temperature, power supply and clock frequency out of operational range, following an erase response triggered before the security properties cannot be ensured any longer. Please note that:
    - Vddcore and Vddcpu monitoring is deactivated in Stop, Standby and Vbat modes
    - Vdd supply, used by IWDG1 and its clock source, is monitored in all modes excluding Standby or Vbat mode
    - Backup domain supply monitoring is available in all system modes, including Vbat mode
- Detection of unauthorized modification of sensitive data stored in fuses
- Prevention of leakage of information through electro-magnetic emissions and power consumption when using AES algorithm (in SAES) or private key cryptography (in PKA)
- Detection of physical penetration attempts using passive or active tamper pins (e.g. using meshes)
- Side channel analysis countermeasures, protecting AES computations (in SAES) and private key cryptography (in PKA).

3.3.6 ~~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.~~

Non-conformance rationale:

This feature is not available to the users. It is implemented but not accessible.

Indeed, the TOE is certified with all debug features disabled out of reset. This debug protection can be frozen anytime until next reset by setting the DENREG bit in BSEC peripheral. ROM code in the TOE is not setting this bit.

As described in section 1.4.2, the JTAG or SWD interface remains enabled (under reset only) to inject the RMA password that can switch the device to the RMA_LOCK state. In this state OTP secrets stored in words 32 to 95 are hidden, hence the platform is no more functional.

Although not part of the TOE-certified configuration, it is possible to select any JTAG or SWD connection as a source of internal tamper, as described in “anti-tamper” part of section 4.2.1 of [2].

3.4 Additional Security Functional Requirements

The platform fulfills the following security functional requirements:

3.4.1 Cryptographic Operation

The platform provides the application with cryptographic operations such as encryption, decryption, hashing, authentication, signature functionality with a list of algorithms specified in Table 5 for key lengths and modes defined in Table 5.

Informational:

This SFR contributes to meet the PCI requirements B9, B10, B11, B24 and B26 (see [DTR]). Cryptographic algorithms and key sizes are listed in Appendix E of [DTR].

Conformance rationale:

The Platform provides to applications the cryptographic algorithms, modes of operation and minimum/maximum key size described in Table 5. Side channel resistance is described in notes.

For more details on side channel resistant cryptographic algorithms, refer to “Hardware-accelerated cryptography” part of Section 4.2.1 in [2]. For more information on hashing algorithms please refer to Section 34 in [3].

Some of those algorithms are used by the boot ROM.

Table 5. TOE cryptographic operations

Operations	Algorithm	Specification	Key lengths	Modes
Encryption, decryption	AES(1)	FIPS PUB 197 NIST SP800-38A	128, 192, 256 bits	ECB, CBC, CTR
Authenticated encryption or decryption		NIST SP800-38C NIST SP800-38D		GCM, CCM
Cipher-based message authentication code		NIST SP800-38D		GMAC
Protected modular exponentiation (signature, decryption, key agreement...)	RSA(2)	IETF RFC 8017 NIST SP800-56B FIPS PUB 186-4	Up to 4096 bits	RSA 1024, 2048, 3072, 4096
Signature	ECDSA	ANSI X9.62 IETF RFC 7027 FIPS PUB 186-4 SEC 1, SEC 2(3)	Up to 640 bits	Nist: P256, P384, P521 Brainpool: bp256r1, bp384r1, bp512r1 SEC 2(3): secp256k1, secp256r1, secp384r1, secp521r1
ECC scalar multiplication (public key generation, key agreement, shared secret generation...)	ECDH ECIES	ANSI X9.42 ANSI X9.63 SEC 1, SEC 2(3)		
Cryptographic hash	SHA-2(4)	FIPS PUB 180-4	N/A	SHA2-224, SHA2-256, SHA2-384, SHA2-512
	SHA-3(4)	FIPS PUB 202	N/A	SHA3-224, SHA3-256, SHA3-384, SHA3-512. SHAKE128, SHAKE256

(1) AES algorithm with key sizes of 128 and 256 bits (and not DES/TDES) can run accelerated with side-channel attack resistance in SAES peripheral.

(2) Other operations not written in this table (like RSA CRT exponentiation or ECDSA signature verification) are not protected against side channel attacks.

(3) Standards for Efficient Cryptography, https://www.secg.org

(4) These algorithms must not be used when manipulating sensitive information.

3.4.2 Cryptographic KeyStore

The platform provides the application with a way to store secret keys such that not even the application can compromise the confidentiality of this data. This data can be used for the cryptographic operations encryption, decryption, authenticated encryption/ decryption.

Informational:

This SFR contributes to meet the PCI requirements B9 and B18 (see [DTR]).

Conformance rationale:

The Platform provides hardware mechanisms to protect the integrity and confidentiality of AES 128 or 256-bit keys in the KeyStore, thanks to encryption using a key derived from the device hardware unique key (HUK). Resulting decrypted keys are automatically stored in write-only key registers, without disclosing any clear-key data to the application. Additionally, if application tries to overwrite part of the key, the whole key is erased. The key derived from the HUK is never disclosed to the MPU and is only accessible by the SCA-protected AES hardware crypto engine (SAES peripheral). The key derived from the HUK is different if it used by a secure code or a non-secure code.

Application-defined 256-bit BHK, stored in tamper-protected backup registers, can be XORed with DHUK when encrypting KeyStore items. This way KeyStore cannot be decrypted until BHK is re-installed by secure boot code.

DHUK and BHK are only usable in side-channel attacks resistant SAES peripheral.

For more details, refer to “Cryptographic key storage” parts of Section 4.2.1 in [2].

4 Mapping and Sufficiency Rationales

4.1 SESIP3 Sufficiency

Table 6 SESIP3 Sufficiency

ASE: Security Target evaluation	ASE_INT.1 ST Introduction	Section 1	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”.
	ASE_OBJ.1 Security requirements for the operational environment	Section 2	The objectives for the operational environment in “Security Objectives for the Operational Environment” refer to the guidance documents.
	ASE_REQ.3 Listed Security requirements	Section 3.3 to Section 3.4	All SFRs in this ST are taken from [1]. “Verification of Platform Identity” is included. “~~Secure Update of Platform~~” is not included (justification in ALC_FLR.2).
	ASE_TSS.1 TOE Summary Specification	Section 3	All SFRs are listed per definition, and for each SFR the implementation and verification are defined in “Security Functional Requirements”.
ADV: Development	ADV_FSP.4 Complete functional specification	Section 1.3, and material provided to the evaluator	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.
	ADV_IMP.3 Complete mapping of the implementation representation of the TSF to the SFRs	Material provided to the evaluator	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.
AGD: Guidance documents	AGD_OPE.1 Operational user guidance	Section 1.3	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.
	AGD_PRE.1 Preparative procedures	Section 1.3	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.
ALC: Life-cycle support	ALC_CMC.1 Labelling of the TOE	Section 1.3	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.
	ALC_CMS.1 TOE CM Coverage	Section 5, and material provided to the evaluator	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.
	ALC_FLR.2 Flaw reporting procedures	Section 3.2	The flaw reporting and remediation procedure is described.
ATE: Tests	ATE_IND.1 Independent testing: conformance	Material provided to the evaluator	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.
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.

5 Documentation references

Table 7. References

Reference	Definition
Evaluation Documents	
[1]	Security Evaluation Standard for IoT Platforms (SESIP), version 1.1 (June 2021), GlobalPlatform, GP_FST_070
[SR]	PIN Transaction Security (PTS), Point of Interaction (POI), Modular Security Requirements, version 6.0 (June 2020), PCI Security Standards Council LLC
[DTS]	PIN Transaction Security (PTS), Point of Interaction (POI), Modular Derived Test Requirements, version 6.0 (June 2020), PCI Security Standards Council LLC
Developers Documents	
[2]	UM2885, STM32MP13xx Security Guidance, version 2, STMicroelectronics
[3]	RM0475, Reference manual STM32MP13xx advanced Arm®-based 32-bit MPUs, version 0.5, STMicroelectronics
[4]	UM2237, STM32CubeProgrammer software description user manual, revision 18, STMicroelectronics
Standards	
[5]	SESIP Profile for Secure MCUs and MPUs, version 1.0 (Oct 2021), GlobalPlatform, GPT_SPE_150

Glossary

Table 8. Glossary

Term	Definition
Boot hardware key	256-bit AES encryption or decryption key stored in backup registers, erased in case of tamper, and not readable nor writable after boot by the application (dedicated key bus to SAES).
Hardware unique key	The device embeds two Hardware unique keys: DHUK and RHUK.

Abbreviations

Table 9. Abbreviations

Term	Definition
BHK	Boot hardware key
DHUK	Derived HUK
HUK	Hardware unique key
PCI	Payment card industry
PKA	Public key accelerator
POI	Point of interaction
POS	Point of sales
PTS	Pin transaction security
RHUK	Root HUK

Revision history

Table 10. Document revision history

Date	Revision	Changes
24 Nov 2020	1	- Cut 1.0 certification release
23 Sep 2022	2	- Cut 1.2 certification release candidate 1 - Update to 3.4.1 and 3.4.2
24 Nov 2022	3	- Updated version for security guidance
16 Dec 2022	4	- Security target is now inspired by GlobalPlatform Protection Profile (not compliant) - Cut 1.2 certification release candidate 2
2 Jan 2023	4b	- Removed HMAC from the TOE
11 Jan 2023	5	- Reduced security target scope, closer to revision 1 - Revision history cleaned up - Cut 1.2 certification release candidate 3
24 Jan 2023	5b	- Alignment with UM2885 rev2 (Jan 23)
30 Jan 2023	5c	Platform name changed from “Integrated circuit: STM32MP13xx (DieID= 0x501)” to “STM32MP13xx”

Contents

List of tables

List of figures