EFR32MG22 Wireless Gecko SoC Family

SESIP Security Target

Version: 0.8
Date: 24/03/2021

1 Version history

Version	Date	Description	Author
0.1	23/09/2020	First draft	Silicon Labs
0.2	08/10/2020	Update from internal review	Silicon Labs
0.3	29/10/2020	Update from evaluation review	Silicon Labs
0.4	25/11/2020	Update from evaluation review	Silicon Labs
0.5	30/11/2020	Update from evaluation review	Silicon Labs
0.6	15/01/2021	Update to address action items and EM1 feedback	Silicon Labs
0.7	05/02/2021	Addressed feedback from the certifier	Silicon Labs
0.8	24/03/2021	Added software attacker resistance SFRs to the scope	Silicon Labs

Table 1 ST version

Contents

2 Identification

2.1 Identification of this ST

EFR32MG22 Wireless Gecko SoC Family, SESIP Security Target, Version: see Table 1, Date: see Table 1.

SESIP version is 1.0, see [SESIP].

This Security Target claims **strict** conformance to [Profile] in line with [SESIP]. This SESIP Profile describes the requirements sufficient to fulfil the hardware requirements for a chip of the DTSec requirements as described in [DTSEC-PP]. The claims to [Profile] are focused on the Base-PP SFRs, which ensure the execution of platform trusted code, and in particular functions related to Secure Boot, Updatability and Debugging.

On top of the above, this ST demonstrates readiness to specific DTSec requirements as described in section 5.3, complemented with a rationale in Appendix A on how to implement the full set of functional requirements described in [Profile].

2.2 Identification of the platform

Reference	Value	Verification method described in
Commercial name	EFR32MG22	[DATA-SHEET]
HW reference	EFR32MG22C224F512	[DATA-SHEET]
HW Version	Die rev C	[DATA-SHEET]
FW name	ROM code OTP code VSE root code	[PROD-PROG] [GECKO-SDK]
FW version	ROM: rev C (tied to HW Version) OTP: v7 VSE: 1.2.7	[PROD-PROG] [DATA-SHEET]
mbedTLS	2.6.1	3.3.2 [SIMPLICITY-REF]
SE_Manager	3.0 (as part of Gecko SDK Suite)	[GECKO-SDK]

Table 2 Platform identification

2.3 Identification of the guidance

2.3.1 Manuals

For the full list of the Platform guidance documentation refer to section 7.1.

3 Platform Overview

The EFR32 product family (henceforth the Platform) combines an energy-friendly MCU with a high performance radio transceiver. The devices are well suited for secure connected IoT multi-protocol devices requiring high performance and low energy consumption.

The Platform is an SoC that consists of an MCU and its associated firmware (ROM code, OTP flash and VSE firmware in flash) that provides the functionality defined in this document. The Platform also includes the external SDK libraries mbedTLS and SE_Manager, providing user-friendly interfaces to access the security functions related to cryptography, secure debug and secure update. No non-Platform components are required to run the platform defined in this document.

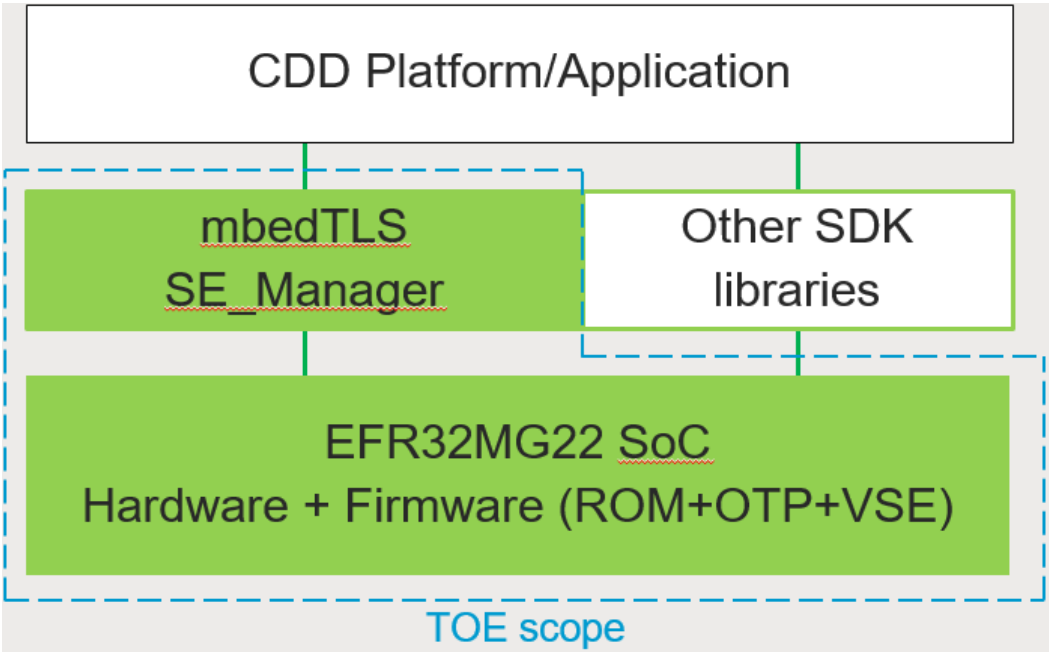

Figure 1 Platform scope

Figure 1 above shows the scope of the Platform, which is composed of the Hardware platform (with corresponding firmware) and the mbedTLS and SE_Manager SDK libraries.

Further details of the different components of the hardware are shown in Figure 2 below:

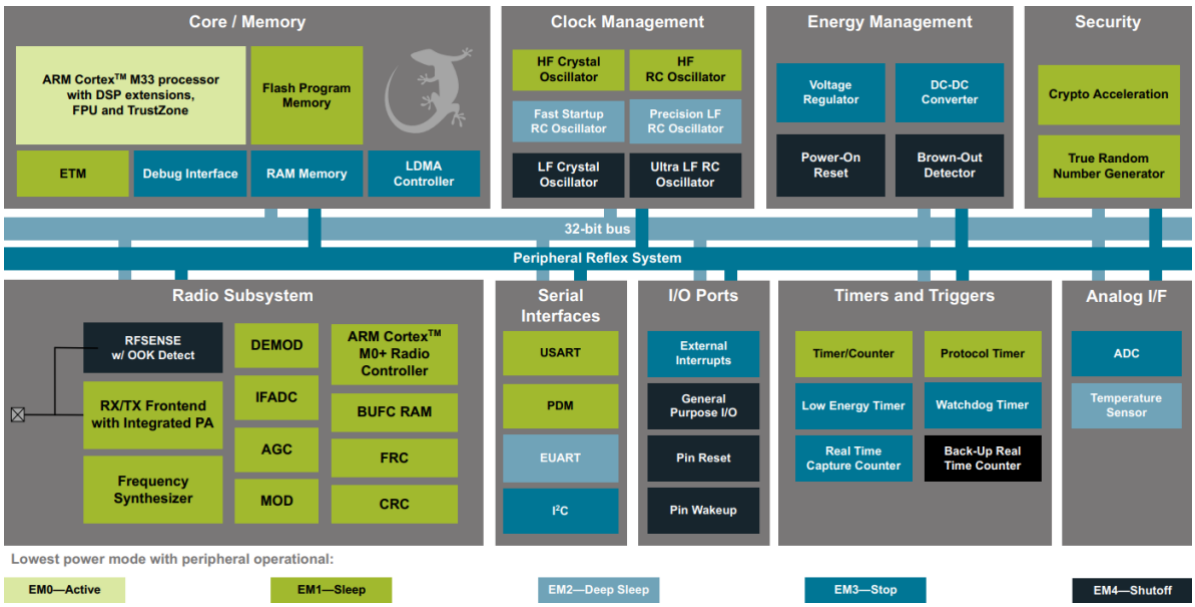

Figure 2 EFR32MG22 System-On-Chip Block Diagram

The Platform is intended to be used as a **base** to develop a fully featured DTSec platform with CDD (Connected Diabetes Device) application that complies with [Profile].

As the Platform is only an SoC and not a full DTSec platform with CDD application, it is only partially compliant to [Profile].

The main security features of the Platform are:

- Secure boot, to control the platform's initialization sequence using the SoC services.
- Secure debugging in case of investigation need using the SoC services.
- Secure update for platform life cycle handling using the SoC services by means of the SE_Manager.
- Cryptographic support using CRYPTOACC by means of mbedTLS API.
- Random number generation using CRYPTOACC by means of mbedTLS API.

These security features are covered directly by the security requirements defined in section 5 Security Requirements of this document.

The Platform also provides additional functional features, not related to the claimed security functionality. These features are summarized below:

- High performance radio transceiver
    - Low power consumption in transmit, receive, and standby modes
    - Excellent receiver performance, including sensitivity, selectivity, and blocking
    - Excellent transmitter performance, including programmable output power, low phase noise, and power-amplifier (PA) ramping
    - Ultra-low energy RF detection for wake-up from any energy mode, through RFSENSE
- Configurable protocol support, including standards and customer-developed protocols

- Preamble and frame synchronization insertion in transmit, and recovery in receive
- Flexible CRC support, including configurable polynomial and multiple CRCs for single data frames
- Basic address filtering performed in hardware

- High performance, low power MCU system
  - High Performance 32-bit ARM Cortex-M33 CPU
  - Flexible and efficient energy management
  - Complete set of digital peripherals
  - Peripheral Reflex System (PRS)
  - Precision analog peripherals
- Low external component count
  - Fully integrated 2.4 GHz BALUN
  - Integrated tunable crystal loading capacitors

3.1 Physical Scope

The physical scope includes the hardware and firmware identified in 2.2 Identification of the platform and the guidance identified in 2.3 Identification of the guidance.

3.2 Logical Scope

The logical scope comprises the secure functions defined in section 3 above.

4 Security objectives for the operational environment

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

- The operating system or application code are expected to verify the correct version of all platform components it depends on, as described in section 9.3 of [DATA-SHEET], in [REF-MAN] section 6.4, and section 5.2.2 of [PROD-PROG].
- The integrator is expected to make use of the *Secure Boot with RTSL* feature as described in the [SEC-BOOT].
- The integrator is expected to configure the debug functionality as described in [SEC-DEBUG]and [PROD-PROG] to meet the extra physical attacker resistance.
- The operational environment must not allow the deployment of untrusted code (as described in [SEC-BOOT]). That means that all code running on the product is known to the product vendor and the product vendor can confirm that the code cannot harm the claimed security functionalities.
- The CDD application is expected not to use SHA-1 unless for backward compatibility reason and security impact and implication are analyzed and acceptable for the use case.

5 Security Requirements

5.1 Security Assurance Requirements

The claimed assurance package is SESIP3 as defined in [SESIP] section 4.3.

5.1.1 Flaw remediation process (ALC_FLR.2)

Silicon Labs has a Product Security Incident Response Process to intake hardware and software vulnerabilities, triage such issues, remediate them where possible, and communicate the vulnerabilities and recommendations to security researchers and product stakeholders. This plan is described in internal documents [FLR] and [FLR-INTERNAL].

Instructions for researchers to disclose vulnerabilities to Silicon Labs are located at the following URL: https://www.silabs.com/security/product-security

The method described recommends the researcher or other party encrypt the email using the Silicon Labs-supplied PSRIT PGP Key, and to address the encrypted email to product-security@silabs.com.

The email will be received by a member of the Product Security Incidence Response Team, who will create a case in an internal ticket tracking system. The ticket will be assigned to a PSIRT team member who is responsible for triaging the issue and working with internal R&D teams to prioritize mitigation and communication efforts.

The case owner is also responsible for direct communication and coordination with the researcher/discloser. If the PSIRT team determines the issue should be shared publicly, a Security Advisory will be drafted and published on our security portal.

Security researchers and other stakeholders can subscribe to receive security advisories via the security portal. Instructions can be found here: https://www.silabs.com/security

If the vulnerability is located in stack code or another software component, the patch will be delivered via an SDK update that is published via Simplicity Studio.

5.2 Base-PP Mandatory Security Functional Requirements

5.2.1 Verification of Platform Identity

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

Conformance rationale: the version of the SoC platform can be verified visually via the method described in [DATA-SHEET] sections 7.3, 8.3, 9.3, and in [REF-MAN] section 6.4, and

VSE Firmware can be verified using Simplicity Commander via the method described in section 5.2.2 of [PROD-PROG].

ROM version is tied to HW version.

The OTP version can be determined by manufacturing and date code on device top mark as described in [DATA-SHEET] sections 7.3, 8.3, 9.3.

5.2.2 Secure update of platform

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

Conformance rationale: the SoC platform's Firmware can be updated to a newer version through the Virtual Secure Element Upgrade functionality as described in section 5 of [UG-BOOT].

The delivery of the VSE Firmware payload may be performed via the debug interface or via an over-the-air upgrade operation. In both cases the signed encrypted and versioned SE Firmware payload is loaded into on-chip flash memory and a VSE mailbox command `sl_se_apply_se_image()` is executed containing a pointer to the VSE Firmware payload. This command terminates in a reset event, which causes the CPU to enter supervisory mode and execute the system boot process.

On boot, the VSE root code recognizes the pending VSE firmware upgrade command, authenticates the VSE Firmware payload signature against the Silicon Labs public key in ROM and validates that the firmware version of the payload is higher than the installed VSE Firmware. If authentication and version checks pass, it will decrypt and replace the VSE Firmware with the contents of the VSE Firmware payload image. If either authentication or version checks fail, the upgrade will be aborted and an error will be returned in the VSE mailbox status register.

5.2.3 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 an infinite loop.

Conformance rationale: the platform authenticity and integrity is ensured by the secure boot of the platform which is performed as described in section 1.3 of [SEC-BOOT]. The secure state consists of an infinite loop that persists until the Platform is reset.

When the device is released from reset, the CPU assumes root mode privileges, and code execution begins from root code in ROM memory. The ROM code validates the signature of the VSE Firmware in flash against a Silicon Labs NIST P-256 public key that is also stored in ROM. If this authentication fails, the platform will go to an infinite loop. If authentication passes, program control passes to the VSE Firmware. This authentication step cannot be bypassed.

The VSE Firmware will check the state of the Secure Boot setting in OTP memory. If Secure Boot of the application is enabled, the VSE Firmware will perform an authenticity check of the

user application in flash memory. The signature of the user application is validated against a NIST P-256 public Sign key that is stored in immutable OTP memory. If this authentication fails, the platform will go to an infinite loop. If authentication passes, the CPU will transition to user mode and program control will pass to the user application.

5.2.4 Secure Debugging

The platform only provides ***debug access port when debug unlock is enabled*** authenticated as specified in ***section 6.3 of [SEC-DEBUG]*** with debug functionality.

The platform ensures that all data stored by the application, with the exception of ***none***, is made unavailable.

Conformance rationale*:* the Platform provides secure debug access to authenticated users. The authentication process is based on a challenge-response scheme based on ECDSA-P256-SHA256.

If Secure Debug is enabled on the platform and the debug port has been locked, then all application data is made unavailable. From this state, the debug port can be unlocked by presenting the debug port with an authenticated debug unlock token. If the token is successfully authenticated, the debug port will be unlocked and remain unlocked until the next pin or power-on reset event, at which point it will return to the locked state. If the token authentication is unsuccessful, the debug port will remain locked.

One of the provisioning steps of Secure Debug is for the user to program a NIST P-256 public Command key into immutable OTP memory. This public key is used to verify the authenticity of the debug unlock token. Other device attributes that participate in the secure debug unlock token are the serial number of the device and a 128-bit random Challenge that is generated by the device and stored in secure NVM memory.

The detailed construction of the authenticated debug unlock token is complex and described in section 6.3 of [SEC-DEBUG]. What follows is an abbreviated description that contains the relevant security elements.

The debug unlock token consists of 3 parts: the unlock command, the Debug Access Certificate, and a Challenge Signature. The Debug Access Certificate and the Challenge Signature are security objects.

The Debug Access Certificate contains a list of operations the certificate is authorized to perform such as debug unlock or disable tamper, a NIST P-256 public Certificate key, and the serial number of the device to be unlocked. The Debug Access Certificate is signed by the private key associated with the Command key in OTP memory.

The Challenge Signature provides a method to revoke a previously-issued authenticated debug unlock token. The Challenge Signature is constructed by signing an object containing the Challenge stored in NVM memory and a few other items using the Certificate private key.

When the debug port receives an authenticated debug unlock token, it verifies the signature on the Debug Access Certificate against the Command public key in OTP memory and then verifies the Challenge Signature with the Certificate public key from the Debug Access

Certificate, along with matching the issued command to the allowed operations specified in the Debug Access Certificate.

Previously issued debug unlock tokens can be rendered invalid by issuing a command to the device to roll its Challenge value, in which the device will generate a new 128-bit random Challenge using its TRNG and replace the old value in NVM memory. If a previous unlock token is presented to the device after the Challenge has been rolled, the authentication will fail the Challenge Signature authentication step.

5.3 DTSec–PP Security Functional Requirements (profile)

As mentioned in 3 Platform Overview, the Platform implements a subset of the requirements defined in [Profile].

5.3.1 Cryptographic Operation

5.3.1.1 AES

The platform provides the application with ***encryption and decryption*** functionality with ***AES*** as specified in ***[FIPS197]*** for key lengths ***128, 192 or 256 bits*** and modes ***ECB, CTR, CBC, CFB, GCM, CBC-MAC, GMAC, CCM***.

5.3.1.2 Hash

The platform provides the application with ***hashing*** functionality with ***SHA*** as specified in ***[FIPS180-4]*** for key lengths ***160 bits (SHA-1), 224 bits (SHA-224) or 256 bits (SHA-256)*** and modes ***not applicable***.

Note: as stated in section 4 of this document, the use of SHA-1 is discouraged and only offered for backward compatibility with legacy systems.

5.3.1.3 ECDSA

The platform provides the application with ***signature generation and verification*** functionality with ***ECDSA*** as specified in ***[FIPS800-56A]*** for key lengths ***192 or 256 bits*** and modes ***not applicable***.

5.3.1.4 ECDH

The platform provides the application with ***Diffie-Hellman key derivation*** functionality with ***ECDH*** as specified in ***[FIPS800-56A]*** for key lengths ***192 or 256 bits*** and modes ***not applicable***.

Conformance rationale*:* the Platform provides support for the aforementioned algorithms, key sizes and modes, as described in section 11 of [REF-MAN]. The platform uses the following **mbedTLS** functions to access the cryptographic functionality:

Algorithm	Standard	Cryptographic operation	mbedTLS call(s)
AES	[FIPS197]	Encryption and Decryption	`mbedtls_cipher_setup()`
SHA	[FIPS180-4]	Hashing	`mbedtls_md_setup()`
ECDSA	[FIPS800-56A]	Signature generation	`mbedtls_ecdsa_sign()`
		Signature verification	`mbedtls_ecdsa_verify()`
ECDH	[FIPS800-56A]	DH key derivation	`mbedtls_ecdh_setup()` `mbedtls_ecdh_calc_secret()`

5.3.2 Cryptographic Random Number Generation

The platform provides the application with a way based on ***thermal noise source*** to generate random numbers to as specified in ***[AIS31] class PTG3***.

Conformance rationale: the True Random Number Generator on the platform is a non-deterministic random number generator that harvests entropy from a thermal energy source. The platform uses `mbedtls_ctr_drbg_random()` function which results in a TRNG by means of CRYPTOACC as defined in 11.5 [REF-MAN].

5.4 Optional SFRs commonly added

5.4.1 Limited Physical Attacker Resistance

The platform detects or prevents attacks by an attacker with physical access before the attacker compromises ***the secure initialization of the platform and the secure debugging***.

Conformance rationale: the Platform implements physical attack countermeasures on the secure boot and secure debug functionalities. The implemented countermeasures include: enabling voltage detectors, and fault injection code hardening.

5.4.2 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.

Conformance rationale: Access to VSE code, data, and hardware functions is disabled via hardware before the application is allowed to execute. The device must be reset to re-enable access to VSE resources which ensuring that access to VSE resources is never present when the application is executing.

The root mode is the isolation mechanism that ensures an effective isolation between the user application and the platform. The root mode is only available to the bootloader and is

deactivated when the control passes to the user application. VSE RAM memory and flash areas are cleared and are no longer accessible when in user mode.

5.4.3 Software Attacker Resistance: Isolation of Platform Parts

The platform provides isolation between platform parts, such that an attacker able to run code in ***VSE RootMode*** can compromise neither the integrity and confidentiality of the ***ROM Root of Trust*** nor the provision of any other security functional requirements.

Conformance rationale: The ROM Root of Trust contains no confidential information and thus confidentiality is not at issue. The nature of the public key and firmware contained in ROM is such that its integrity cannot be compromised (behavior cannot be edited) even if an attacker were to gain the ability to execute code while VSE Root Mode is still enabled. Thus, ROM and OTP bootloader are stored in immutable regions and cannot be modified.

VSE code is signed using a manufacturer key not accessible to the user application. Therefore, there is an effective separation between the user application and the VSE regions. Furthermore, root mode is protecting the bootloader, and code cannot be disclosed or executed and data cannot be disclosed when the user application is running.

6 Mapping and sufficiency rationales

6.1 Assurance

Assurance Class	Assurance Families	Covered by	Rationale
ASE: Security Target evaluation	ASE_INT.1 ST Introduction	Section 2 Identification Section 3 Platform Overview	ST reference in section 2.1 Platform reference in section 2.2 Platform overview in section 3
	ASE_OBJ.1 Security requirements for the operational environment	Section 4 Security objectives for the operational environment	The objectives for the operational environment refers to the guidance documents.
	ASE_REQ.3 Listed Security requirements	Section 5 Security Requirements	All SFRs in the profile are taken from [SESIP]. "Verification of Platform Identity" is included. "Secure update of platform" is included.
	ASE_TSS.1 Platform Summary Specification	Section 5 Security Requirements	Each SFR in section 5 includes a conformance rationale
ADV: Development	ADV_FSP.4 Complete functional specification	Functional specification conformed of the guidance documents listed in 7.1 Guidance documentation.	The evaluator will validate suitability of the provided evidence.
	ADV_IMP.3 Complete mapping of the implementation	Full source code provided to the evaluators.	The evaluator will validate suitability of

	representation of the TSF to the SFRs		the provided evidence.
AGD: Guidance documents	AGD_OPE.1 Operational user guidance	Guidance documentation as listed in 7.1 Guidance documentation	The evaluator will validate suitability of the provided evidence.
	AGD_PRE.1 Preparative procedures	Guidance documentation as listed in 7.1 Guidance documentation	The evaluator will validate suitability of the provided evidence.
ALC: Life-cycle support	ALC_CMC.1 Labelling of the TOE	Configuration list as listed in 7.1 Guidance documentation	The evaluator will validate suitability of the provided evidence.
	ALC_CMS.1 TOE CM Coverage	Configuration list as listed in 7.1 Guidance documentation	The evaluator will validate suitability of the provided evidence.
	ALC_FLR.2 Flaw reporting procedures	Section 5.1.1 Flaw remediation process (ALC_FLR.2) Flaw remediation documents [FLR] and [FLR-INTERNAL].	The flaw remediation procedures are described.
ATE: Tests	ATE_IND.1 Independent testing: conformance	N/A	The evaluator will perform independent testing.
AVA: Vulnerability Assessment	AVA_VAN.3 Focused vulnerability analysis	N/A	The evaluator will perform penetration testing.

Table 3 Assurance rationale

7 References

7.1 Guidance documentation

[REF-MAN]	EFR32xG22 Wireless Gecko Reference Manual, Revision 1.0, August 2020
[DATA-SHEET]	EFR32MG22 Wireless Gecko SoC Family Data Sheet, Revision 1.0, June 2020
[SEC-BOOT]	AN1218: Series 2 Secure Boot with RTSL, Revision 0.3, July 2020
[UG-BOOT]	UG266: Silicon Labs Gecko Bootloader User's Guide, Rev. 1.4
[SEC-DEBUG]	AN1190: Series 2 Secure Debug, Revision 0.3, September 2020
[ERRATA]	EFR32MG22 Errata, rev 0.4
[PROD-PROG]	AN1222: Production Programming of Series 2 Devices, rev 0.3
[SIMPLICITY-REF]	UG162: Simplicity Commander Reference Guide, rev 1.9
[CI-LIST]	EFR32MG22 Configuration item list, v0.1
[FLR]	PS1012 – Security Vulnerability Disclosure Policy, Rev B
[FLR-INTERNAL]	CRISIS006 - Product Security Incident Response plan (PSIRP), Rev E
[GECKO-SDK]	Gecko SDK Suite, v3.0, July 2020
[AGD]	EFR32MG22 Wireless Gecko SoC Family Certified Product Guidance, version 0.3, 15/01/2021

7.2 SESIP documentation

[Profile]	SESIP profile for DTSec Connected Diabetes Devices, SESIP-PP-DTSEC, version 1.0 draft, February 2021
[DTSEC-PP]	Protection Profile for Connected Diabetes Devices, v2.0, November 25, 2017
[SESIP]	GlobalPlatform Technology Security Evaluation Standard for IoT Platforms (SESIP), GP_FST_070, Version 1.0, Dated March 2020

7.3 Standards

[AIS31]	A proposal for: Functionality classes for random number generators, Version 2.0, 18 September 2011.
[FIPS197]	FIPS PUB 197: Advanced Encryption Standard (AES), 26 November 2001.
[FIPS180-4]	FIPS PUB 180-4: Secure Hash Standard (SHS), August 2015.
[FIPS800-56A]	NIST Special Publication 800-56A Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography, Rev. 3, April 2018.

7.4 Terms and definitions

BGM	Blood Glucose Monitor
BG reading	Blood Glucose data acquired by the CDD device hardware.

CDD	Connected Diabetes Device
CRC	Cyclic Redundancy Check
DTSec	Diabetes Technology Society cybersecurity standard for connected diabetes devices
SFR	Security Functional Requirement

Appendix A: DTSec compliance rationale

A.1 DTSec PP requirements

The Platform is intended to be used as base to develop a DTSec platform with CDD (Connected Diabetes Device) application that complies with [Profile].

As the Platform is only an SoC and not a full DTSec platform with CDD application, it is only partially compliant to [Profile].

The following table, extracted from [Profile], shows which requirements from [Profile] are met by the Platform (marked in ***bold italics***) and which requirements are left to be implemented by the CDD platform/application developer:

[DTSEC-PP] Functional Requirement	Covered by SESIP [Profile]	Rationale
FTP_ITC.1	Secure Communication Support Secure Communication Enforcement	Full coverage by direct translation.
FIA_NET_EXT.1.1	None	This is a typical requirement for the CDD device, rather than the DTSec platform alone.
FDP_IFC.1		
FDP_IFF.1		
FDP_DAU.1	Secure Storage	The CDD will process BG readings and/or store them. This requirement ensures that these readings cannot be modified while stored without this being noticed.
FPT_TST_EXT.1	***Secure Initialization of Platform***	For the purpose of meeting DTSec, this checking must be extended to the CDD Application as well. The platform must therefore offer this functionality to the CDD Developer and describe how to do this in his Guidance.
FCS_COP.1	***Cryptographic Operation***, iterated for: • Communication • ***Platform***, Application and BG Readings Integrity Cryptographic Key Generation	Full coverage by direct translation.
FCS_COP_EXT.1	***Cryptographic Random Number Generation***	Full coverage by direct translation.
FIA_AFL.1	Attestation of platform genuineness	

(optional)	Attestation of Application Genuineness Secure Communication Support ***Secure Debugging***	The DTSec platform will require subjects to be authenticated in order to perform security relevant actions, if the optional SFRs need to be implemented.
FIA_UAU.1 (optional)		
FIA_UAU.6 (optional)		
FPT_PHP.3 (optional)	Physical Attacker Resistance ***Limited Physical Attacker Resistance***	Depending on the nature of the DTSec platform, the Platform may provide a different level of physical protection.

A.2 DTSec Profile additional objectives

When the Platform is used as basis for a full DTSec product to comply with [Profile], the final product shall include also the following objectives for the operational environment:

- The CDD developer shall ensure that their application mandates that explicit user interaction takes place (e.g. pressing a button) before the application calls a pairing function.
- The CDD developer shall define the information flow policy and correctly configure the device to ensure that this policy is implemented. The CDD developer should also make sure that the data coming out of the BLE connection that goes into his application is carefully sanitised before processing the data so that it cannot corrupt his application, by, for example:
    - The data is not longer than the application expects, possibly causing buffer overflows
    - The data is not shorter than expected
    - The data does not contain values the application does not understand (control characters, end-of-string markers in the middle of a string etc.)
    - The data does not contain values that the application understands but may not be able to handle (e.g. choosing menu option #7 when there are only 6 menu options) for boundary conditions
- The CDD developer needs to secure the integrity and authenticity of the BG readings.
- The CDD developer needs to follow the corresponding guidance documentation on how to extend the checking of the platform authenticity and integrity to the CDD Application.

A.3 DTSec Profile additional security features

When the Platform is used as basis for a full DTSec CDD product aiming to comply with [Profile], the final product shall include also the security features:

- Secure Communication
- Secure Storage of sensitive data