Qualcomm Technologies, Inc.

Qualcomm SPU260 Security Target Lite

80-NU430-10 Rev. AC

September 9, 2022

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Revision history

Revision	Date	Description
AC	September 9, 2022	Added TSS rationale
AB	August 19, 2022	Editorial updates
AA	July 29, 2022	Initial official release

Contents

Figures

Tables

1 Introduction to SPU260 Security Target

This Security Target is defined for the Qualcomm® Secure Processor Unit (SPU260), supported by Trusted Management Engine (TME), embedded in the SM8450 host System-on-Chip (SoC) combined with a double data rate (DDR) memory in a package-on-package (PoP) configuration and its corresponding IC dedicated software and associated documentation.

The hardware of TOE is comprised of the SPU and the TME subsystems. The IC dedicated software is comprised of a SPU Firmware, SPU Software, TME Firmware and TME Software, as described in 3.2 . The TOE is delivered together with the appropriate guidance documentation.

This Security Target includes claims derived from the Security IC Platform Protection Profile with Augmentation Packages *[ICPP]*.

Because the Target of Evaluation (TOE) is not a usual smartcard IC, additional security functions of the hardware and the operating system (OS) have been added to this Security Target.

1.1 Security Target reference

"Qualcomm SPU260 Security Target Lite, 80-NU430-10 Rev. AC, Qualcomm Technologies, Inc."

1.2 TOE reference

The TOE described in this Security Target is named *"Qualcomm Secure Processor Unit SPU260 (Version: 5.0)* in SM8450 SoC (Qualcomm® Snapdragon™ 8 Gen 1) with TME (Version: 1.0.1) and symmetric and asymmetric crypto support".

1.3 Conventions

The security functional requirements (SFRs) have the following conventions:

- Assignments are underlined
- Selections are marked with squared brackets [ ]
- Refinements are written in *italic*
- Iterations are identified with an extension of the Security Functional Requirement name

For example, the requirement in the Standard is written as follows:

> The TSF shall perform [assignment: list of cryptographic operations] in accordance with a specified cryptographic algorithm [assignment: cryptographic algorithm] and

cryptographic key sizes [assignment: cryptographic key sizes] that meet the following: [assignment: list of standards].

In this document, the requirement is written as follows:

The TSF shall perform message authentication code generation in accordance with a specified cryptographic algorithm CMAC using AES and cryptographic key sizes 128 bits, 256 bits that meet the following: FIPS 197 and NIST SP 800-38B.

1.4 Technical assistance

For assistance or clarification on information in this document, open a technical support case at https://support.qualcomm.com/.

You will need to register for a Qualcomm ID account and your company must have support enabled to access our Case system.

Other systems and support resources are listed on https://qualcomm.com/support.

If you need further assistance, you can send an email to qualcomm.support@qti.qualcomm.com.

2 Target of Evaluation overview

2.1 Target of Evaluation

The TOE is an integrated Secure Element, composed by two subsystems namely Secure Processor Unit (SPU) and TME, which are integrated in the SoC within a stacked DDR package on SoC package (package form factor is non-TOE). It is designed as a tamperproof device providing secure storage and a secure execution environment for processing of sensitive data and for performing cryptographic operations using protected keys stored in its secure storage. Secure Elements can be used for multiple application areas that require a high level of security. Examples are as follows:

- User authentication and password storage
- Content protection
- Payment
- Subscriber Identity Module (SIM)
- Storage and management of digital identities
- Secure key storage
- Root of trust
- Secure Storage of sensitive user data (for example health care records)

The TOE has dedicated interfaces to other components of the SoC, which allow those components to communicate with the TOE and request services from the TOE.

The TOE is comprised of a hardware layer and IC dedicated software providing interfaces for application developers.

The TOE will allow for dedicated applications to execute on the OS of the TOE to provide security services as listed above. Those applications are not part of the TOE, but the TOE OS provides services to verify the integrity and authenticity of such applications using digital signatures.

The TOE communicates with the other components of the SoC either using shared memory or using shared Configuration and Status Registers (SP-sCSR), interrupts, and power control messages.

The hardware of the TOE is internally structured into three main units:

- The Secure Central Processing unit, which performs the general operations of the TSF.
- The Crypto Management unit, which performs the cryptographic operations and generates, manages, and protects keys.
- The TME subsystem unit, of which the involvement in the TOE is limited to support the SPU during the secure boot process.

For a more detailed description of those units, their functions and how they are internally structured, see section *3.1* on TOE definition.

2.2 Non-TOE hardware and software

2.2.1 Non-TOE Software

The TOE is embedded onto Qualcomm Snapdragon™ 8 Gen 1, used in mobile applications. Snapdragon™ 8 Gen 1 comprises a High Level Operating system (HLOS, such as Android) that is required for the TOE to boot and properly communicate with the rest of the Hardware and Software. Only this way the TOE is in the certified configuration.

2.2.2 Non-TOE Hardware

The TOE requires presence of the host SoC (Snapdragon™ 8 Gen 1), DDR (external RAM), sMMU as well as NVM to be functional. The final device in the field consists of a PoP. One package contains the SoC integrating the TOE hardware and the other package contains the DDR. The TOE software image is stored encrypted in NVM. DDR is referred as an external RAM, which is required to load and execute the SPU software on the SPU.

2.3 Security functions

2.3.1 Internal Security functions

The TOE implements the following internal Security functions:

- Access control to the various memories (OTP, RAM, ROM) and peripherals
- Access control to keys managed in hardware through enforcement of key policy
- Secure boot and secure loading of TOE software stored outside the TOE using the TOE root of trust (ROM code and TME subsystem)
- Protection of User Data stored outside the TOE
- Secure loading of user applications stored outside the TOE
- Secure update mechanism of the TOE software or applications

- Domain separation between applications executed by the TOE (for both user and system applications)
- Anti-replay island and software freshness protection

2.3.2 Cryptographic services (API)

Note: This section is an overview. See functional security requirements descriptions for details on each algorithm and key type/size.

The TOE provides cryptographic services using the support of the Crypto Management Unit. Services provided through the API for user applications are as follows:

- Generation of random numbers (used for key generation)
- Secure key storage providing the possibility to have keys stored in the SP-CMU that are not readable by the SP-CPU. The SP-CPU can only request to perform cryptographic operations using those keys.
- Secure key generation and zeroization
- Symmetric encryption and decryption using the following:
  - AES with 128 bit and 256 bit keys
  - TDES with 112 bit and 168 bit keys
- Hash functions: SHA-1, SHA-256, SHA-384, SHA-512
- HMAC using keys up to 512 bit length and using SHA-1, SHA-256, SHA-384 or SHA-512
- CMAC with AES using 128 bit and 256 bit keys
- Asymmetric cryptographic operations:
  - RSA 1024 bit and 2048 bit
  - Elliptic curves cryptography with NIST P-192/224/256/384/521, Brainpool P-192/224/256/320/384/512 non-twisted (r1) and Curve25519 curves.

2.3.3 Physical protection

The TOE provides a number of functions and features that are designed to counter physical attacks. Those include the following:

- Memory scrambling/memory encryption
- Side-channel analysis countermeasures
- Fault attacks sensors and countermeasures
- Memory/registers integrity checking

3 TOE description

3.1 TOE boundary and interface

The TOE are two independent subsystems that will be integrated in the SM8450 SoC in a manner that is agnostic to the TOE hardware and IC dedicated software implementation details. The TOE serves as an independent Root of Trust within the SoC. It does not rely on any external entity for any security enforcement, allowing it to be evaluated as a separate entity. Even though NVM, DDR and package form factor are considered non-TOE hardware, they do not enforce any security functionality but the TOE relies on them for TOE functionality. The TOE has also its own ROM code and a dedicated TME subsystem for secure boot operations (other TME functions different than those related to secure boot operations are considered SFR-non-interfering).

The TOE and its hardware interfaces to the SoC into which it is integrated are shown in the following figure.

The TOE Hardware interfaces are:

- Battery Monitor (indicates when power is lost)
- System Access to allow SPU to read/write in the SPU shared memory in DDR (SP-sMEM) as well as in SPU protected memory partitions in DDR and non-volatile memory (NVM). This interface is also used by the SPU to communicate with other peripherals of the SoC such as GPIO.
- QSB AXI master between TME and System NOC to enable the communication between the TME subsystem and the SPU for secure boot operations.
- QHB_Slave between TME and Config NOC.
- Interface to read/write SP-sCSR with the SoC. These CSRs are among other things used as doorbell for communication between the SPU and the rest of the SoC and in particular with the TME for secure boot operations.
- Interface to the RPM (Resource and Power Manager) module to provide SPU power requirement and for receiving time data from RPM.
- Interface to the Clock Controller (GCC) to provide clock requirement and receive external clock and reset signal for the entire chip, including the SPU.

As indicated earlier, the TOE provides a TSF to cryptographically protect User Data before storage in an SPU DDR partition or SPU NVM partition. This TSF also ensures that User Data read from these locations are trustworthy before processing them internally.

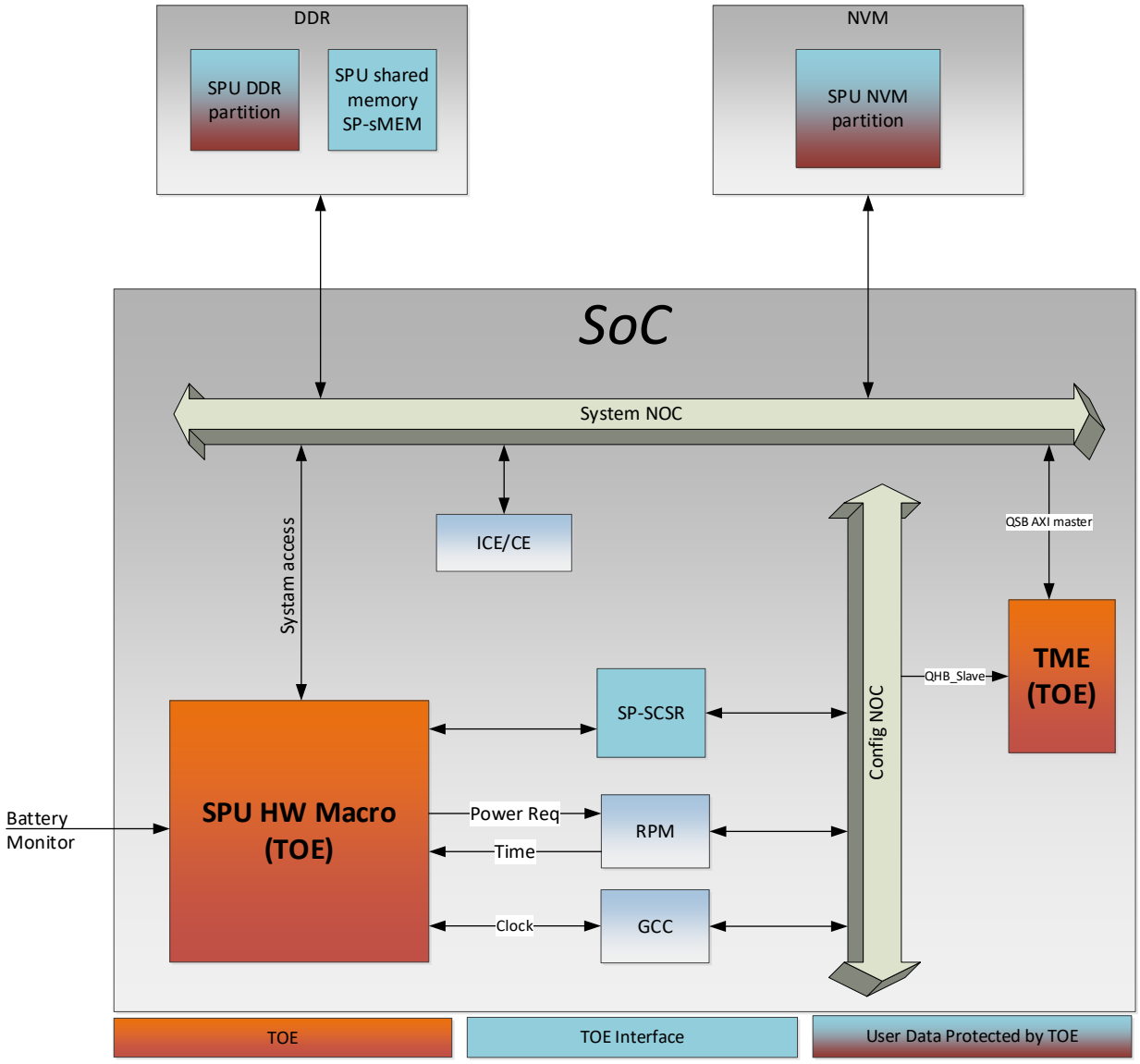

Figure 3-1 TOE components and their interfaces to SoC

The TOE Software interface consist in Application Programming interfaces providing:

- Communication services
- External Memory storage (read/write) services
- Cryptographic services

The TOE communicates with the other components of the SoC via the SPU shared memory, the SPU shared Configuration and Status Registers.

3.2 Scope of the TOE

3.2.1 Overview

The TOE is composed of:

- SPU Hardware: a hardmacro synthetized independently of rest of SoC and integrated as a black box
- SPU IC dedicated software: is a superset of code formed by SPU Firmware (ROM code) and SPU Software (RAM code loaded from NVM). The code belonging to OS and CryptoLib is formed by different pieces of these both sources.
  - SPU Firmware:
    - Primary Boot Loader (PBL): includes part of the low level cryptography of the CryptoLib and is used to load the encrypted software part of the SPU after being enabled by TME, as described in section *3.2.3.*
    - Mission ROM: Part of OS, which includes drivers and another part of the low level cryptography of the CryptoLib.
  - SPU Software:
    - MCP: Part of OS, which provides APIs and services for the SPU system applications and the user applications.
    - SPU System Applications: Part of OS, which provides additional TOE functionalities that are not packaged in the SPU Firmware or MCP.
    - User applications: They are considered as Security IC Embedded Software, and therefore out of the scope of the TOE. Note that the user applications can only access HW features via the APIs and services from MCP and therefore does not have direct interaction with firmware.
- TME Hardware: a separated subsystem integrated in the SoC but outside the SPU which communicates with the SPU during secure boot process.
- TME IC dedicated software: as part of the TOE IC dedicated software and a superset of code formed by TME Firmware (ROM code) and TME Software (RAM code loaded from NVM). It is in charge of giving support to the SPU in the early stages of the secure boot process (other functions of the TME are considered as SFR-non-interfering). Its parts are the following:
  - TME Firmware:
    - TME Mission ROM: stored in TME CPU ROM, includes drivers and low level cryptography.
    - TME Sequencer Firmware: stored in TME Sequencer ROM, which verifies the TME Sequencer Software in TME Sequencer RAM prior to the TME Sequencer Software is executed.

- TME PBL: stored in TME CPU ROM, includes part of the low level cryptography and is used to authenticate the TME Core Software image prior to the TME Core Software is executed.

- TME Software
  - TME Sequencer Software: manages the communication with the registers and clocks.
  - TME Core Software: supports the secure boot of the SPU.

The following figure shows the TOE IC dedicated software components of the TOE as well as the Application Programming Interface of the TOE as is used by the user application running in the TOE. Note that in Figure 3-2, SPU System Application and User Application are referred as S.APP and U.APP, respectively.

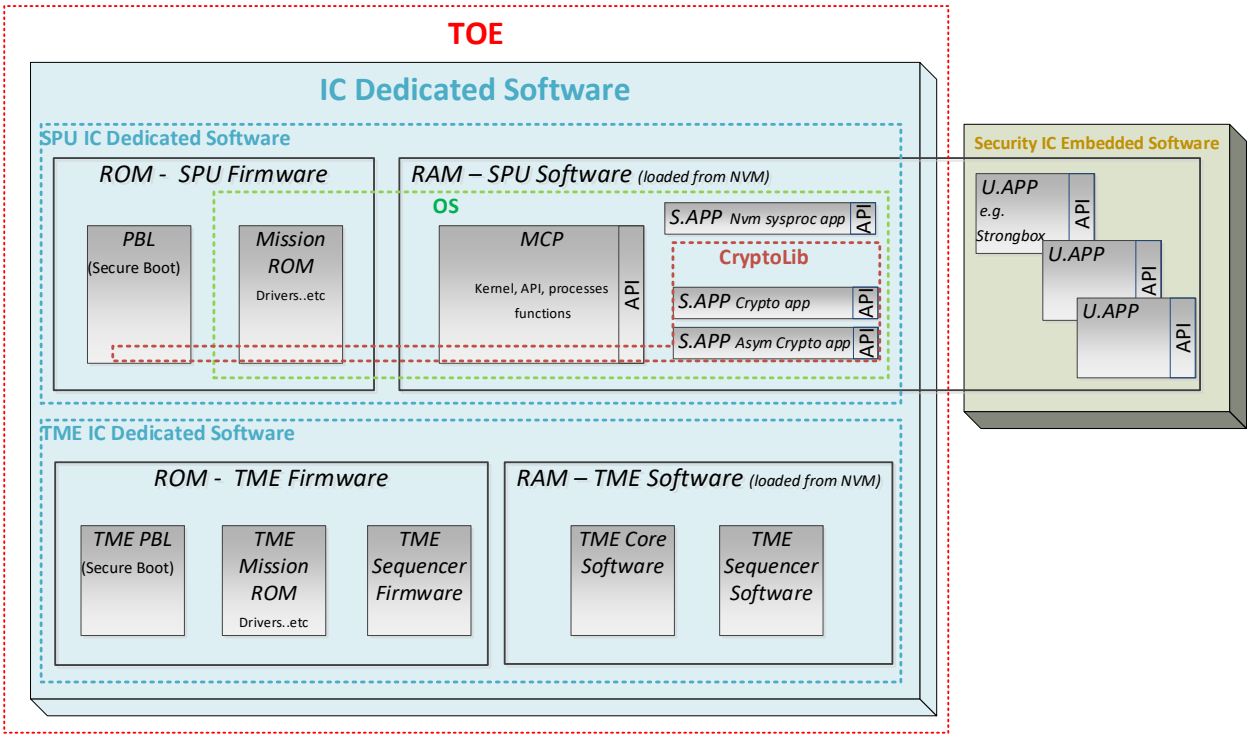

Figure 3-2 TOE IC dedicated software components

The OS is composed of MCP, system applications (cryptoapp, asym_cryptoapp, nvm_sysproc) and Mission ROM. The CryptoLib is composed of cryptography relevant part of PBL, cryptography relevant part of Mission ROM, cryptography relevant part of MCP and system applications (cryptoapp and asym_cryptoapp).

3.2.2 Hardware

The physical boundary of the TOE includes the TOE hardware which consists of the following components:

- For SPU subsystem (SPSS)

- Secure Central Processing Unit (SP-CPU), which executes the main code of the SPU.
- Memory: SP-RAM (for kernel, applications stacks) and SP-ROM (for the PBL image and mission ROM).
- Memory Protection Unit (SP-MPU), which is responsible for controlling access to memory (SP-ROM and SP-RAM) and the DMA controller (SP-DMA)
- Crypto Management Unit (SP-CMU), which provides support for random number generation and key generation (SP-RNG) as well as HW accelerated hash, symmetric and asymmetric cryptographic functions (SP-CE and SP-PKA). It also holds a Crypto Management Controller (SP-CMC) in which Key Table (SP-KT) exists.
- Processor Interconnect Bus, which is used for data exchange between the SP-CPU and the SP-CMU.
- Local Resource Manager (SP-LRM), which provides the interface to the Clock Source, the reset line and the interface to the Sensors (voltage, temperature, frequency etc.).
- Security Controller (SP-SC) Block, which holds the Qualcomm Fuse-Programmable Read-Only Memory (SP-QFPROM) area including keys that the SP-CPU can request to be used by the SP-CMU but cannot read directly.
- SP-Timer and SP-Watchdog, used to provide timer functionality for the TOE independent from other timers within the SoC.
- The always-on Island that contains the part of the Anti-Replay controller (SP-AR) and the Always-On Timer (SP-AOTimer).
- The External Memory Manager (SP-ExtMM) providing read/write capabilities to TOE external memory.

- For TME subsystem (TMESS)
  - TME CPU, hardened CPU to execute the TME PBL, TME Mission ROM and TME Core Software. It is also the entry point into TME (slave) and includes the AXI master to the SNoC, with address remappers.
  - TME CPU ROM and TME CPU RAM, memories used by TME CPU.
  - TME Sequencer, module to execute TME Sequencer Firmware and TME Sequencer Software which controls all the resources under TME HW layer.
  - TME Sequencer ROM and TME Sequencer RAM, memories used by TME Sequencer.
  - TME XPU-C, module containing the access control protection units.
  - TME PRNG, module for hardware-only random number generator.
  - TME Debug, debug module for debugging purposes.
  - TME HSDMA, High Speed DMA with FIFOs.
  - TME Key Table, which Store keys used by the crypto engines on TME.
  - TME Hardware CSR, register map used by the CPU or JTAG to execute commands on the sequencer.

- TME Fuse controller, module which contains the QFPROM.
- TME Asymmetric Crypto, which provides ECC & RSA based asymmetric cryptography services to TME CPU and SoC through the SEQ.
- TME Symmetric Crypto, which performs symmetric cryptographic operations for the sequencer.
- TME Clock Controller, module which implements the control of the clocks.
- SCSR XPU4, module which controls the access to SCSR.

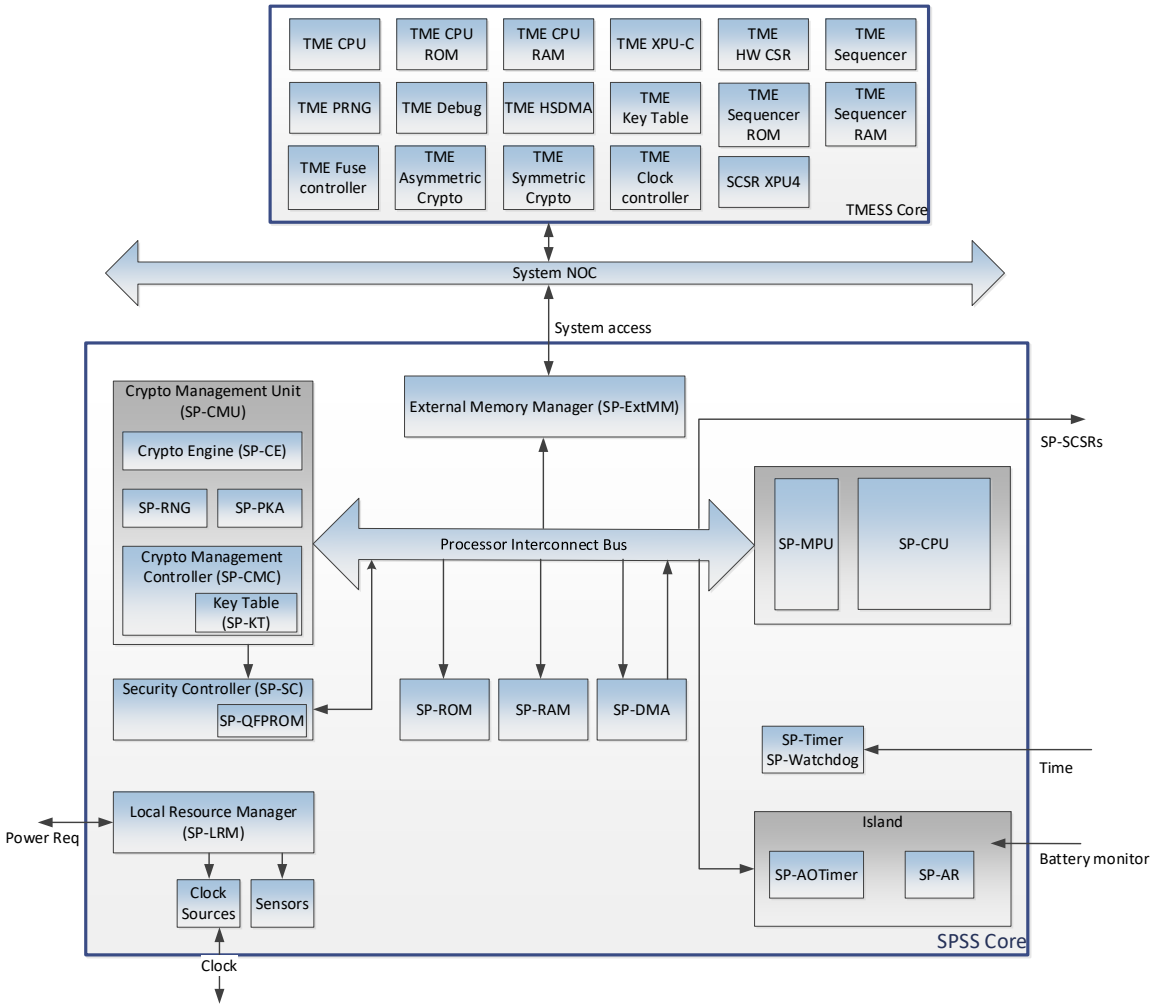

Figure 3-3 TOE hardware components

3.2.2.1 SPU subsystem

3.2.2.1.1 SP-CPU

The SP-CPU is the main processing unit of the SPU. It executes the general firmware and software of the TOE. The SP-CPU provides memory protection that allows the implementation of a secure OS that separates unprivileged applications from each other and allows protection of critical resources from direct access by applications without using the OS services that control access to such critical resources.

3.2.2.1.2 SP-MPU

The memory protection unit of the SPU (SP-MPU) is responsible for controlling access to memory (SP-ROM and SP-RAM) and the DMA controller (SP-DMA) in accordance with the access control attributes. It is programmed by the SP-CPU from its privileged mode, allowing the OS running on the SP-CPU to protect memory areas from direct access by applications and protect memory areas assigned to one application from direct access by another application. The SP-MPU is supplemented by two permission checkers embedded in the SP-RAM and SP-ROM.

3.2.2.1.3 SP-CMU

The SP-CMU is a separate subsystem within the SPU that is responsible for the cryptographic operations performed by the SPU as well as the generation and protection of key material used for those operations. The SP-CMU subsystem actually acts like a separate hardware security module in a general purpose operating environment. It consists of the following:

- Crypto engine (SP-CE) as the central processing unit of the SP-CMU (which also includes the hardware implementation of the cryptographic coprocessors: AES, SHA-1 and SHA-256).
- Random number generation unit (SP-RNG), which consists of two physical noise sources and a DRBG.
- Hardware support for accelerating asymmetric crypto operations (SP-PKA).
- Crypto Management Controller (SP-CMC), which manages the Key Table (SP-KT) including the transfer of keys to the SP-KT.

The SP-CMU is programmed by the SP-CPU, which can request operations such as key generation, loading the key, or performing cryptographic operations; the SP-CPU does not have access to the keys themselves that are managed by the SP-CMC (unless the key attributes allow the key to be exported in clear outside the SP-CMU subsystem). The SP-CMC manages the keys stored in the SP-KT, which are the keys private to the SPU.

3.2.2.1.4 SP-SC

The security control component (SP-SC) contains the SP-QFPROM (also called as OTP) and is responsible for controlling access to OTP areas there. This includes protection of areas that are write-protected and control access of the SP-CPU to allow it to only access dedicated OTP items that are not indicated as read-protected. Some secret data stored in OTP are stored in encrypted form.

3.2.2.1.5 SP-LRM

The SP-LRM is responsible for interrupt handling and management of the SPU. The sensors that detect operational problems, faults, or potential attacks are connected to the SP-LRM and cause an interrupt when they detect a problem. The SP-LRM passes the interrupt to the SP-CPU for handling and requires the SP-CPU to clear the interrupt, indicating that it has received and processed the interrupt. Alternatively, the SP-LRM can be configured to perform a cold reset upon specific sensors detection.

Note that internal interrupts of the SP-CPU (such as system tick time expiration, SP-MPU permission error, and privilege exception) are handled directly by the SP-CPU and not by the SP-LRM, but an interrupt caused by the SP-Timer is handled by the SP-LRM.

3.2.2.2 TME subsystem

3.2.2.2.1 TME Subsystem Core

The TME subsystem is a separated part of the TOE but within the SoC and in charge of the early stages of the secure boot process. It is composed by the TME CPU which runs the TME PBL, TME Core Firmware and TME Core Software in its own RAM and the TME Sequencer, which runs the TME Sequencer Firmware and TME Sequencer Software. The TME Sequencer drives the steps during the secure boot while the TME CPU is finally in charge of bringing the SPU out of reset in this process. Only these modules related to secure boot are considered involved in the security functionality as SFR-supporting.

3.2.3 Firmware, Software and Application

3.2.3.1 Related to SPU

The SPU-PBL and OS with the software API are considered as logical boundary of the TOE to user application. The OS manages the access to the services provided by the TOE, implements software countermeasures and controls the user applications.

The TOE OS consists of:

- The Mission ROM (stored in SP-ROM)
- The software (loaded from NVM and stored in SP-RAM and DDR)
  - MCP image (in SP-RAM)
  - System applications (in DDR)

- cryptoapp
- asym_cryptoapp
- nvm_sysproc

In a nutshell, the SPU-PBL in SPU Firmware loads MCP which loads system applications.

The TOE contains the firmware in SP-ROM that is used for the secure boot process (supported by TME). In addition, part of the SPU Firmware contains drivers that are used in operational mode by the loaded SPU Software (after the secure boot).

The TOE also contains the SPU Software, MCP, which is stored, signed and encrypted in external non-volatile memory and loaded into SP-RAM at runtime by the PBL. The PBL verifies the signature and decrypts the software each time before the MCP is loaded and executed by the SP-CPU.

The OS is formed by the MCP, the system applications and the Mission ROM. The OS is running on the SP-CPU and provides services to user applications loaded for the SP-CPU. The OS verifies the integrity and authenticity and enforces confidentiality of any applications loaded to the TOE including system applications.

The MCP image and the Applications are stored in external memory and can be updated by downloading a newer version in the external memory. A set of rollback counters prevent the TOE from loading an older version of MCP or an application.

The OS is also responsible to separate applications executing on it from each other and control that an application uses only those services and objects it is supposed to use (as defined in the manifest of the downloaded and signed application package). This OS is part of the TOE and implements some TSF.

The system applications, as part of OS, are stored in external non-volatile memory and are loaded into DDR at runtime by the aforementioned MCP. The TOE contains three system applications:

- Cryptoapp: implements the RSA key generation service.
- Asym_cryptoapp: implements additional asymmetric cryptography services and the corresponding API.
- Nvm_sysproc: implements nvm system process.

The OS provide the following services to User applications via APIs:

- Cryptographic services (AES, Hashing and message authentication codes) with keys either held as retained keys within the SP-CMU (in the SP-KT) or with keys provided by the application. If retained keys are used, the OS verifies that the application is allowed to use those keys and if they are used in accordance with the key attributes. In addition, the TOE provides random number generation services.
- Cryptographic services (AES, TDES, ECDSA, ECDH and RSA) with keys provided by the application.
- NVM storage for User data. User data is stored (in external DDR/NVM) encrypted, authenticated and protected against replay. The TOE maintains a unique key for each application that is used for these cryptographic operations.
- Communication services with external entities (for example Modem sub-system, HLOS or Trusted Execution Environment).

- Application loading services.

3.2.3.2 Related to TME

The TME PBL, TME Mission ROM, TME Sequencer Firmware, TME Sequencer Software and TME Core Software are considered as logical boundary of the TOE from the side of the TME subsystem.

The logical scope components are:

- TME PBL (stored in TME CPU ROM) which authenticates the TME Core Software image.
- TME Mission ROM (stored in TME CPU ROM) includes drivers and low level cryptography .
- TME Sequencer Firmware (stored in TME Sequencer ROM) which verifies the TME Sequencer Software image.
- TME Sequencer Software (stored in NVM and is loaded to TME Sequencer RAM) which allows access to the underlying hardware.
- TME Core Software, (stored in NVM and is loaded to TME CPU RAM) which is in charge of taking SPU out of reset.

3.2.4 Package

The final device in the field consists of a PoP. One package contains the SoC integrating the TOE hardware and the other package contains the DDR that is needed for SPU to be operational and in a certified configuration. The package is part of TOE operational environment.

The package in which the TOE is delivered is shown in an abstracted way, not to scale and without all connections in the following figure. The figure illustrates highlighted in dark, the physical boundary of the TOE within the final PoP package, not included in the TOE.

PoP solution consists in:

- The Package A containing the SoC bare die integrating the TOE hardware.
- The Package B containing the DDR bare die required for the system to work.

The SoC is integrated into Package A during phase 4 (see *3.2.10*).

The package B is stacked on Package A during phase 6 (see *3.2.10*).

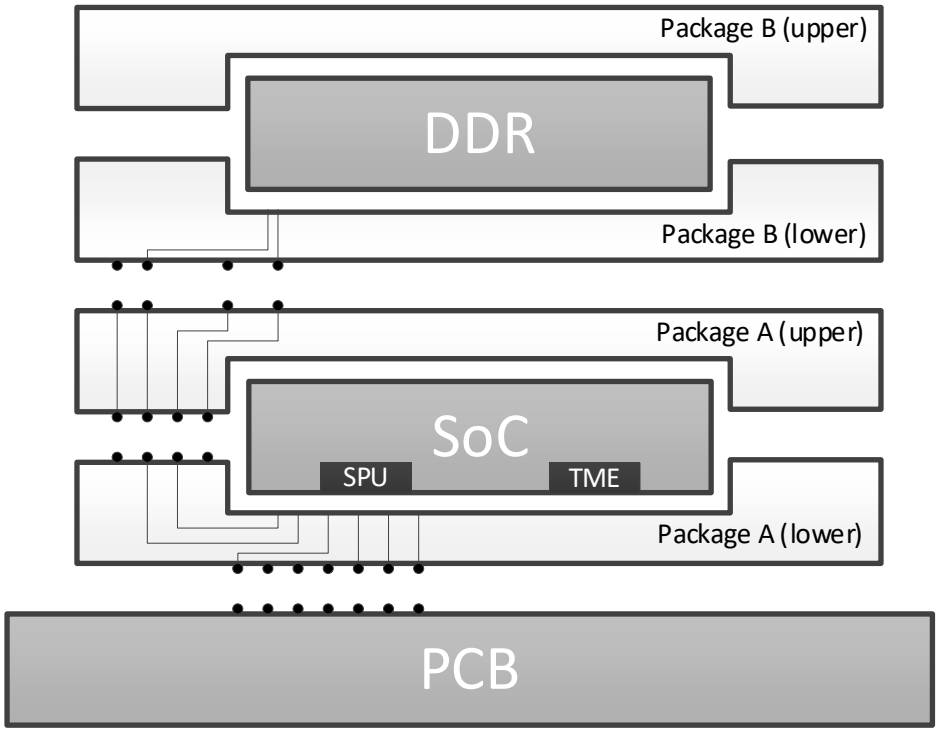

Figure 3-4 TOE package

3.2.5 Guidance documentation

The API for the use of the Services of the Secure Processor as well as associated guidance are provided with the software development kit..

3.2.6 Forms of delivery

The TOE comprises all items that listed in *Table 3-1*.

The TOE hardware and firmware including the personalization data (in SP-QFPROM) will be delivered in the form of a partly packaged SoC to the OEMs to integrate the SoC into his devices.

The integration includes a step of adding a DDR package on top of the partly packaged SoC. This is called a PoP.

In addition, the OEMs will receive the TOE software (MCP image, system applications and TME Core Software image) as part of an overall Qualcomm SW package for the SoC. Qualcomm provides customers access to the Agile system that can be used to download the SW package.

The TOE software will be loaded by the OEMs in the device NVM.

Also the guidance documents listed in *Table 3-1* can be downloaded from the database provided in Agile.

Delivery protection for all TOE components is covered by ALC_DEL and ALC_DVS.

3.2.7 TOE configuration

The following table describes the TOE components and the identifiers:

Table 3-1 TOE components and identifiers

Item Type	Item	Identifier	Form of Delivery
Hardware	SPU hard macro embedded in SM8450 SoC	5.0	Die in package A as in Figure 3-4
	TME hard macro embedded in SM8450 SoC	1.0.1	
	Foundry ID embedded in SM8450 SoC	6	N/A
Firmware	SPU ROM code ▪ PBL ▪ Mission ROM	76100000	Included in SPU hard macro ROM
	TME CPU ROM code ▪ TME PBL ▪ TME Mission ROM	Linked to TME hard macro	Included in TME CPU hard macro ROM as part of TME HW
	TME Sequencer ROM code ▪ TME Sequencer Firmware	Linked to TME hard macro	Included in TME Sequencer hard macro ROM as part of TME HW
Software	SPU Software image, which includes MCP and following system applications: ▪ cryptoapp ▪ asym_cryptoapp ▪ nvm_sysproc	SPSS.A1.1.6-00076-WAIPIO-1	Software image encrypted and signed
	TME Sequencer Software image	Build release string ▪ `53 45 51 5F 46 57 5F 52 45 4C 45 41 53 45 5F 42 55 49 4C 44 5F 56 45 52 53 49 4F 4E 5F 53 54 52 49 4E 47 3D 72 37 39` (“SEQ_FW_RELEASE_BUILD_VERSION_STRING=r79” in ASCII)	Software image signed

Item Type	Item	Identifier	Form of Delivery
	TME Core Software image	Build time string ▪ 4D 61 79 20 30 35 20 32 30 32 32 20 61 74 20 31 36 3A 30 32 3A 34 34 (“May 05 2022 at 16:02:44” in ASCII) Version string ▪ 73 73 67 2E 74 6D 65 66 77 2E 31 2E 30 2E 31 2D 30 30 32 39 39 2D 72 65 6C 65 61 73 65 (“ssg.tmefw.1.0.1-00299-release” in ASCII) Chipset string ▪ 57 61 69 70 69 6F (“Waipio” in ASCII)	Software image signed
Document	*Secure Processor Unit (SPU) – Anti-replay Island (ARI) Overview for SM8450*	80-11140-16, Revision AC	PDF
	Qualcomm® Secure Processing Unit Enablement for SM8450/SM8475 Devices – User Guide	80-PV345-150, Revision AE	PDF
	SM8450/SM8475 Security Guidance for Secure Processing Unit Application Developers	80-PV345-152, Revision AD	PDF
	SM8450/SM8475 Secure Boot Enablement	80-PV345-14, Revision AE	PDF
	SM8450/SM8475 Secure Processor Unit SDK – API Reference	80-PV579-11, Revision AC	PDF
	Qualcomm® Snapdragon™ Secure Processing Unit (SPU) Application Development User Guide	80-NU430-7, Revision AB	PDF
	SMT Assembly Guidelines	SM80-P0982-1, Revision E	PDF

3.2.8 TOE initialization

The TOE is provisioned with individual keys and transitions to operational state during Final Test in Outsourced Semiconductor Assembly and Test (OSAT) premises.

The TOE can only boot fully after it has been integrated in a device containing the Software (MCP image and TME Sequencer / Core Software images) during the product integration phase by the OEMs.

After composite product integration and during operational usage the TOE performs the following action upon boot:

- TME bring up SPU
- Life-Cycle control
- SPU260 configuration

- Verify signature and decrypt the MCP image in SP-RAM
- Execute MCP
- Verify signature and decrypt the MCP image to load the system applications in DDR

3.2.9 TOE integration

The TOE is an integrated part of a larger SoC that itself is intended to be integrated into mobile or other devices.

3.2.10 TOE Life-Cycle

The Life-Cycle of the TOE has been modified (refined with additional phase) compared to *[ICPP]* to match our TOE Life-Cycle.

The Life-Cycle control of the TOE ensures that a device in Test mode cannot run the TOE software and limits access to the TOE firmware (to the minimal set of code required to boot).

The Life-Cycle control of the TOE ensures that the device is in Perso mode before TOE Personalization (phase 5) is performed, where initialization and pre-personalization happens. Initialization includes static data provisioning while pre-personalization includes per chip data provisioning (such as keys)

The Life-Cycle control of the TOE ensures that TOE data (stored during phase 5) and user data (stored during phase 7) are protected in Operational mode.

The process of developing and manufacturing a composite product that contains the TOE is shown in the following figure.

The Firmware and Software development is done as part of the TOE development and the Firmware is internally delivered for the integration into the ROM.

The IC packaging phase embeds the SoC into PCBs on both sides that allow the addition of the DDR and the integration into the final product. DDR addition and integration into the final product is done in Phase 6.

TOE personalization includes the provisioning of keys that allow customers to perform further personalization of their SPU applications while the TOE is in Operational mode.

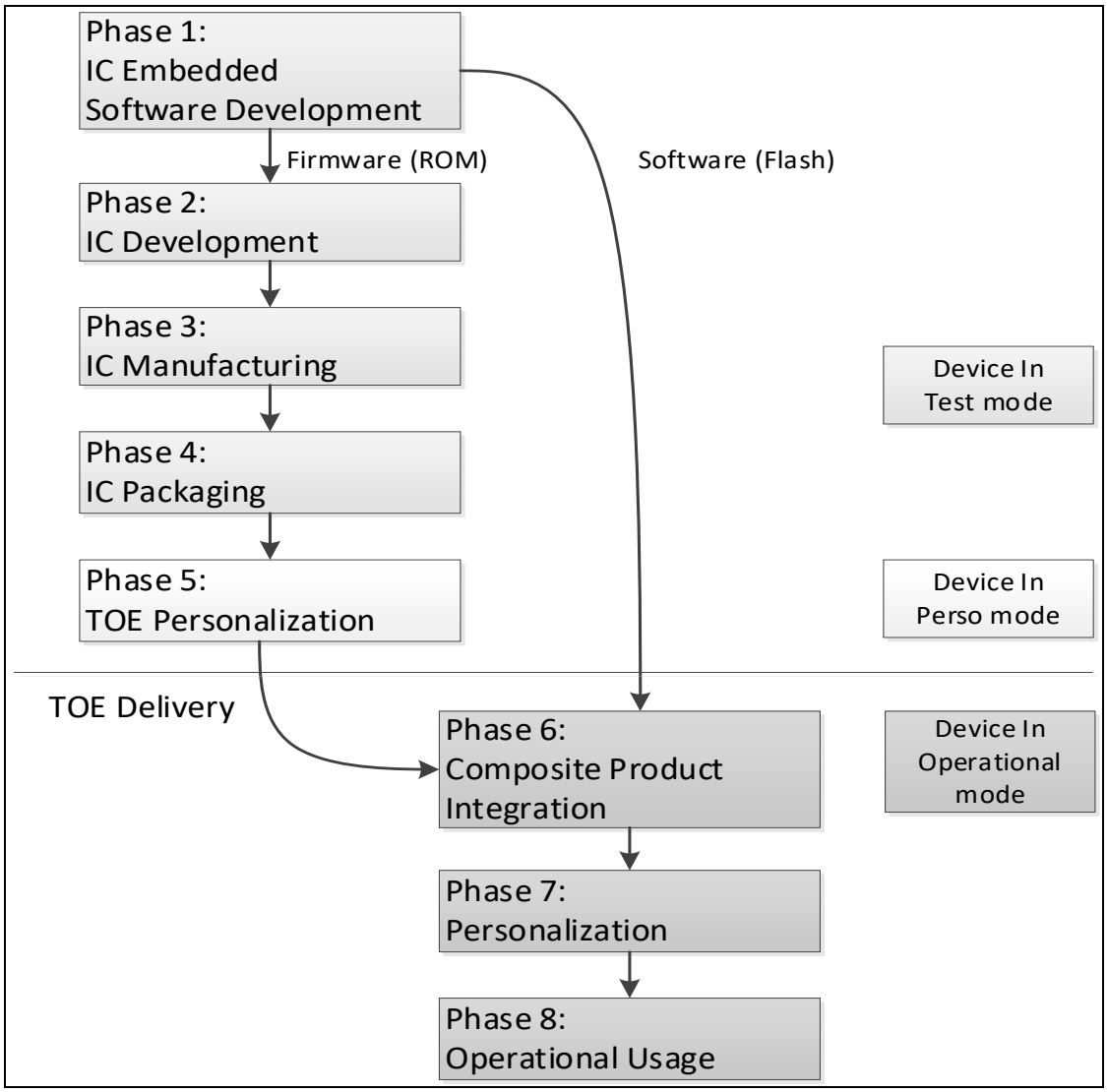

Figure 3-5 TOE Life-Cycle

4 Conformance claims

4.1 CC conformance claims

This Security Target and the TOE claim conformance to *Common Criteria for Information Technology Security Evaluation*, Parts 1, 2, and 3, Version 3.1, Revision 5, which is referred to in this document as *[CC]*.

The Security Target conformance claimed is: Part 1 conformant, Part 2 extended, Part 3 conformant.

4.2 PP claims

This Security Target claims strict conformance to *Security IC Platform Protection Profile with Augmentation Packages*, Version 1.0 (BSI-CC-PP-0084-2014), which is referred to in this document as *[ICPP]*.

4.3 Package claims

The packages for AES and Hash functions from *[ICPP]* have been included.

The Security Target claims conformance to EAL4 augmented by ALC_DVS.2 and AVA_VAN.5.

4.3.1 Conformance Claim Rationale

The TOE specified in this Security Target is a self-sufficient component of a packaged Qualcomm Snapdragon System-on-Chip (SoC). Apart from access to memory, power supply and the connectivity to consumers of the TOE functionality, the remainder of the SoC does not support any aspect of the operation of the TOE.

The TOE can therefore be regarded as a Security Integrated Circuit which implements all functional aspects specified by the protection profile. The TOE provides different types of cryptographic services to external entities, stores and generates keys which can be used for different cryptographic operations. The keys stored and processed by the TOE are protected against logical or physical attacks.

In addition, the development and production Life-Cycle specified by the protection profile is compatible with the one applicable to the TOE.

This allows the conclusion that the protection profile with its intended use cases is applicable to the TOE.

5 Security problem definition

5.1 Definition of assets

The assets to be protected are as follows:

- User data stored inside the SPU, or processed by the SPU
- User data stored outside the TOE while under the control of the TOE
- TSF data used internally by the TOE such as internal encryption keys, SPU Firmware…etc.
- Security IC Embedded Software (user applications), being stored and being executed
- Security services provided by the SPU to the user applications, executed in the SPU

Components of the SoC that are not part of the TOE are considered external entities and they can communicate with the TOE in a similar way an external entity communicates with a smart card. The communication between those components and the SPU is via the SP-sCSR, interrupts, and the shared memory areas. The SP-sCSR act as mailboxes where the other components send requests to the SPU, and the shared memory areas are used for bulk data transfer required to process those requests.

More precisely, the TOE assets that require protection are only in SPU, listed as follows:

- Root keys
- Keys derived from root keys
- Revision
- Life-Cycle state data
- Code execution control
- Code and data integrity, authenticity and confidentiality (load and runtime)
- Debug/Test mode/interface

5.2 Threats

The following threats are defined in *[ICPP]*.

Table 5-1 Security threats

Threat	Description
T.Leak-Inherent	Inherent Information Leakage An attacker may exploit information, which is leaked from the TOE during usage of the Security IC, to disclose confidential user data as part of the assets.
T.Phys-Probing	Physical Probing An attacker may perform physical probing of the TOE (i) to disclose user data while stored in protected memory areas, (ii) to disclose/reconstruct user data while processed, or (iii) to disclose other critical information about the operation of the TOE to enable attacks disclosing or manipulating user data of the Composite TOE or the Security IC Embedded Software.
T.Malfunction	Malfunction due to Environmental Stress An attacker may cause a malfunction of TSF or the Security IC Embedded Software by applying environmental stress to (i) modify security services of the TOE, (ii) modify functions of the Security IC Embedded Software, or (iii) deactivate or affect security mechanisms of the TOE to enable attacks disclosing or manipulating user data of the Composite TOE or the Security IC Embedded Software. This may be achieved by operating the Security IC outside normal operating conditions.
T.Phys-Manipulation	Physical Manipulation An attacker may physically modify the Security IC to (i) modify user data of the Composite TOE, (ii) modify the Security IC Embedded Software, (iii) modify or deactivate security services of the TOE, or (iv) modify security mechanisms of the TOE to enable attacks disclosing or manipulating user data of the Composite TOE or the Security IC Embedded Software.
T.Leak-Forced	Information Leakage An attacker may exploit information, which is leaked from the TOE during usage of the Security IC, to disclose confidential user data of the Composite TOE as part of the assets even if the information leakage is not inherent but caused by the attacker.
T.Abuse-Func	Abuse of Functionality An attacker may use functions of the TOE which may not be used after TOE Delivery to (i) disclose or manipulate user data of the Composite TOE, (ii) manipulate (explore, bypass, deactivate, or change) security services of the TOE, or (iii) manipulate (explore, bypass, deactivate, or change) functions of the Security IC Embedded Software, or (iv) enable an attack disclosing or manipulating user data of the Composite TOE or the Security IC Embedded Software.
T.RND	Deficiency of Random Numbers An attacker may predict or obtain information about random numbers generated by the TOE security service, for instance, because of a lack of entropy of the random numbers provided.
The following threats are not part of *[ICPP]* and have been added:	
T.Boot-Compromise	Compromising the Boot Functionality An attacker might attempt to interfere with the boot process by attempting to boot TSF software not authorized by the TOE.
T.CONFID-TSF-CODE	The attacker executes an application without authorization to disclose the TSF software.

Threat	Description
T.CONFID-APPLI-DATA	The attacker executes an application without authorization to disclose data belonging to another application.
T.CONFID-TSF-DATA	The attacker executes an application without authorization to disclose data belonging to the TSF.
T.INTEG-APPLI-CODE	The attacker executes an application to alter (part of) its own or another application's code.
T.INTEG-TSF-CODE	The attacker executes an application to alter (part of) the TSF software.
T.INTEG-APPLI-DATA	The attacker executes an application to alter (part of) another application's data.
T.INTEG-TSF-DATA	The attacker executes an application to alter (part of) TSF data.
T.AUTH-TSF-DATA	The attacker replaces (part of) TSF data with (part of) TSF data from another device
T.AUTH-APPLI-DATA	The attacker replaces (part of) application data with (part of) application data from another device
T.RBP-TSF-DATA	The attacker performs a replay operation on (part of) TSF data (replay an older version)
T.RBP-APPLI-DATA	The attacker performs a replay operation on (part of) application data (replay an older version)

Threat agents that must be considered are as follows:

- Non-TOE Software executing as nonprivileged software on the SP-CPU, attempting to attack the functionality of the TOE or gain information by observing the behavior of the TOE. Only software whose manifest is vetted by Qualcomm can be loaded into the TOE. Qualcomm vets the software developer and controls privileges of the software by the manifest. To keep the evaluation to a reasonable effort, such software components are not included into the TOE.
- Hardware or software executing on other components of the SoC, attempting to attack the functionality of the TOE or gain information by observing the behavior of the TOE.
- External entities accessing the SoC via its external interfaces.
- External attackers that physically probe the TOE.
- External attackers that attempt to gain access to TSF or user data (critical information) by observing the behavior of the TOE.

5.3 Organizational security policies

The following organizational security policy is defined in *[ICPP]*.

Table 5-2 Organization security policy

Policy	Description
P.Process-TOE	Identification during TOE Development and Production An accurate identification must be established for the TOE. This requires that each instantiation of the TOE carries this unique identification.
P.Crypto-Service	Cryptographic services of the TOE The TOE provides secure hardware based cryptographic services for the IC Embedded Software. Application Note: the following crypto services are supported by hardware: AES, SHA-1, SHA-256, SHA-384, SHA-512, HMAC, CMAC AES and KDF and the asymmetric accelerator for ECDH/ECDSA, RSA_SIGN and RSA_ENC (for all modes, please refer to appendix A).
Two organizational security policies have been added that is not included in *[ICPP]*:	
P.Least-Privilege	Least Privilege for TSF Components The TSF itself is structured into a number of components where some components have their own internal functions and data that is not directly accessible by other components of the TSF. This limits the access from a component to the other component on the SPU.
P.SW_Crypto-Service	SW Cryptographic service of the TOE The TOE provides secure software based cryptographic services for the IC Embedded Software. Application Note: the following crypto service is supported by software: TDES.

5.4 Assumptions

The following assumptions are defined in *[ICPP]*.

Table 5-3 Security assumptions

Assumption	Description
A.Process-Sec-IC	Protection during Packaging, Finishing, and Personalization It is assumed that security procedures are used after delivery of the TOE by the TOE Manufacturer up to delivery to the end-consumer to maintain confidentiality and integrity of the TOE and of its manufacturing and test data (to prevent any possible copy, modification, retention, theft, or unauthorized use). This means that the Phases after TOE Delivery (refer to Sections 1.2.2 and 7.1 in *[ICPP]*) are assumed to be protected appropriately. For a preliminary list of assets to be protected, refer to paragraph 96 (page 29 in *[ICPP]*).
A.Resp-Appl	Treatment of User Data of the Composite TOE All user data of the Composite TOE are owned by Security IC Embedded Software. Therefore, it must be assumed that security relevant user data of the Composite TOE (especially cryptographic keys) are treated by the Security IC Embedded Software as defined for its specific application context.

6 Security objectives

The Security Objectives are taken from *[ICPP]* with some additional security objectives added.

6.1 Security objectives for the TOE

Table 6-1 Security objectives for the TOE

Objective	Description
O.Leak-Inherent	Protection against Inherent Information Leakage The TOE must provide protection against disclosure of confidential data stored and/or processed in the Security IC by measurement and analysis of the shape and amplitude of signals (for example on the power, clock, or I/O lines) and by measurement and analysis of the time between events found by measuring signals (for instance, on the power, clock, or I/O lines).
O.Phys-Probing	Protection against Physical Probing (Refined) Refinement: The TOE must provide protection against disclosure/reconstruction of user data while *transferred or* stored in protected memory areas and processed *by the hardware* or against the disclosure of other critical information about the operation of the TOE. This includes protection against measuring through galvanic contacts, which is direct physical probing on the chips surface except on pads being bonded (using standard tools for measuring voltage and current) or measuring not using galvanic contacts but other types of physical interaction between charges (using tools used in solid-state physics research and IC failure analysis) with a prior reverse-engineering to understand the design and its properties and functions. The TOE must be designed and fabricated so that it requires a high combination of complex equipment, knowledge, skill, and time to derive detailed design information or other information that could be used to compromise security through such a physical attack.
O.Malfunction	Protection against Malfunctions The TOE must ensure its correct operation. The TOE must indicate or prevent its operation outside the normal operating conditions where reliability and secure operation have not been proven or tested. This is to prevent malfunctions. Examples of environmental conditions are voltage, clock frequency, temperature, or external energy fields.

Objective	Description
O.Phys-Manipulation	Protection against Physical Manipulation The TOE must provide protection against manipulation of the TOE (including its software and TSF data), the Security IC Embedded Software, and user data of the Composite TOE. This includes protection against reverse-engineering (understanding the design and its properties and functions), manipulation of the hardware and any data, as well as undetected manipulation of memory contents.
O.Leak-Forced	Protection against Forced Information Leakage ▪ The Security IC must be protected against disclosure of confidential data processed in the Security IC (using methods as described under O.Leak-Inherent) even if the information leakage is not inherent but caused by the attacker by forcing a malfunction (see O.Malfunction: Protection against Malfunctions) and/or ▪ by a physical manipulation (see O.Phys-Manipulation: Protection against Physical Manipulation). If this is not the case, signals that normally do not contain significant information about secrets could become an information channel for a leakage attack.
O.Abuse-Func	Protection against Abuse of Functionality The TOE must prevent that functions of the TOE, which may not be used after TOE Delivery, can be abused to (i) disclose critical user data of the Composite TOE, (ii) manipulate critical user data of the Composite TOE, (iii) manipulate Security IC Embedded Software, or (iv) bypass, deactivate, change, or explore security features or security services of the TOE. Details depend, for instance, on the capabilities of the Test Features provided by the IC Dedicated Test Software, which are not specified here.
O.Identification	TOE Identification The TOE must provide means to store Initialization Data and Pre-personalization Data in its non-volatile memory. The Initialization Data (or parts of it) are used for TOE identification.
O.RND	Random Numbers The TOE will ensure the cryptographic quality of random number generation. For instance, random numbers shall not be predictable and shall have a sufficient entropy. The TOE will ensure that no information about the produced random numbers is available to an attacker because they might be used, for instance, to generate cryptographic keys.
O.AES	Cryptographic service AES The TOE provides secure hardware based cryptographic services for the AES for encryption and decryption.
O.SHA	Cryptographic Service Hash Function The TOE provides secure hardware-based cryptographic services for secure hash calculation.
Security Objectives in addition to the ones defined in *[ICPP]*:	
O.Defense-in-Depth	Defense-In-depth The TOE shall ensure that critical functions and TSF data cannot be accessed by a malicious software executing on the SP-CPU.
O.Secure-Boot	Secure Boot Process The TOE shall ensure that only authorized software is loaded during the boot process after the integrity and authenticity of that software has been verified.
O.SW-TDES	Cryptographic service Triple-DES The TOE provides secure software based cryptographic services implementing the Triple-DES encryption and decryption

Objective	Description
O.RSA	The TOE provides secure cryptographic services implementing the RSA algorithm for signature generation, verification and encryption. This implementation uses dedicated hardware support provided by the TOE.
O.ECDSA	The TOE provides secure cryptographic services implementing the ECDSA algorithm based on the NIST P-192/224/256/384/521 and Brainpool P-192/224/256/320/384/512 non-twisted (r1) for signature generation and verification. This implementation uses dedicated hardware support provided by the TOE.
O.ECDH	The TOE provides secure cryptographic services implementing the ECDH algorithm based on the NIST P-192/224/256/384/521, Brainpool P-192/224/256/320/384/512 non-twisted (r1) and Curve25519 curves for key generation and shared secret generation. This implementation uses dedicated hardware support provided by the TOE.
O.KDF	The TOE provides secure cryptographic services implementing the Key Derivation Function algorithm based on NIST SP 800-108. This implementation uses dedicated hardware support provided by the TOE.
O.HMAC	The TOE provides secure hardware-based cryptographic services for Keyed-Hash Message Authentication Code (HMAC) (FIPS 198-1) calculation.
O.CMAC	The TOE provides secure hardware-based message authentication code generation in accordance with ADVANCED ENCRYPTION STANDARD (AES) (FIPS 197) and the CMAC Mode for Authentication (NIST SP 800-38B).
O.OSData.Access	Restriction of access to data maintained by operating system The TSF must restrict access of any type of software applications running on the SP-CPU exclusively to its own data stored in volatile and non-volatile memory.
O.OSData.Protect	Protection of data maintained by operating system The TSF must protect TSF code, TSF data, application code and application data using cryptographic functions with a cryptographic strength commensurate with the value of the data against loss of confidentiality, integrity, authenticity and against replay.
O.Reallocation	Reallocation of resources The TOE shall ensure that the re-allocation of an internal memory block for the runtime areas maintained by the TOE operating system does not disclose any information that was previously stored.

6.2 Security objectives for the development and operational environment

Table 6-2 Security objectives for the environment

Objective	Description
OE.Process-Sec-IC	Protection during Composite Product Manufacturing Security procedures shall be used after TOE Delivery up to delivery to the end-consumer to maintain confidentiality and integrity of the TOE and of its manufacturing and test data (to prevent any possible copy, modification, retention, theft, or unauthorized use). This means that Phases after TOE Delivery up to the end of Phase 6 (refer to Section 1.2.3 of *[ICPP]*) must be protected appropriately. For a preliminary list of assets to be protected, see section *5.1.*

Objective	Description
OE.Resp-Appl	Treatment of user data of the Composite TOE Security relevant user data of the Composite TOE (especially cryptographic keys) are treated by the Security IC Embedded software as required by the security needs of the specific application context.

6.3 Security objectives rationale

For the threats, assumptions, organizational security policies, and security objectives that are listed in *[ICPP]*, the rationale described there applies. This includes the policy P.Crypto-Service and the included objectives O.AES and O.SHA.

There have been the following threats, assumptions, organizational security policies, and objectives added in this Security Target:

- Threats:
  - T.Boot-Compromise
  - T.CONFID-TSF-CODE
  - T.CONFID-APPLI-DATA
  - T.CONFID-TSF-DATA
  - T.INTEG-APPLI-CODE
  - T.INTEG-TSF-CODE
  - T.INTEG-APPLI-DATA
  - T.INTEG-TSF-DATA
  - T.RBP-TSF-DATA
  - T.RBP-APPLI-DATA
  - T.AUTH-TSF-DATA
  - T.AUTH-APPLI-DATA
- Assumptions:
  - None
- Organizational security policies:
  - P.Least-Privilege
  - P.SW_Crypto-Service
- Security objectives:
  - O.Defense-in-Depth
  - O.Secure-Boot
  - O.SW-TDES
  - O.RSA
  - O.ECDSA
  - O.ECDH

- O.CMAC
- O.HMAC
- O.KDF
- O.OSData.Access
- O.OSData.Protect
- O.Reallocation

The threat T.Boot-Compromise is addressed by the security objective O.Secure-Boot, which requires that the software loaded during the boot process is verified for its authenticity and integrity before it is executed.

The threats T.CONFID-TSF-CODE, T.CONFID-TSF-DATA, T.INTEG-TSF-CODE, T.INTEG-TSF-DATA, T.RBP-TSF-DATA and T.AUTH-TSF-DATA are addressed by the security objective of O.OSData.Protect that requires that TSF data maintained by the operating system is protected with cryptographic mechanisms including encryption, MAC and replay counter to prevent unauthorized disclosure or modification or exchange. The threats are derived from *[JCSPP]*.

The threats T.CONFID-APPLI-DATA and T.INTEG-APPLI-DATA are addressed by the security objective O.OSData.Access limiting applications to access only their own data maintained by the operating system. The threats are derived from *[JCSPP]*.

The threats T.CONFID-APPLI-DATA, T.INTEG-APPLI-CODE, T.INTEG-APPLI-DATA, T.RBP-APPLI-DATA and T.AUTH-APPLI-DATA are addressed by the security objective O.OSData.Protect requires that application data maintained by the operating system is protected with cryptographic mechanisms to prevent unauthorized access. The threats are derived from *[JCSPP]*.

The threat of T.CONFID-APPLI-DATA, T.INTEG-APPLI-DATA, T.CONFID-TSF-DATA, and T.INTEG-TSF-DATA are addressed by the security objective O.Reallocation requiring the operating system to clear internal memory resources upon reallocation. The threats and objective are derived from *[JCSPP]*.

The organizational security policy P.Least-Privilege is addressed by the additional security objective O.Defense-in-Depth, which requires that critical resources are protected by more than just one ring of protection.

The organizational security policy P.Crypto-Service is used as intended in *[ICPP]*. P.Crypto-Service implements a number of cryptographic functions supported by the hardware platform (O.RSA, O.ECDSA, O.ECDH, O.HMAC, O.CMAC, O.KDF).

The organizational security policy P.SW_Crypto-Service implements a cryptographic function implemented in Software (O.SW-TDES).

7 Extended component definition

This Security Target uses the extended components defined in *[ICPP]*:

- FCS_RNG.1
- FMT_LIM.1
- FMT_LIM.2
- FAU_SAS.1
- FDP_SDC.1

For the complete specification and justification of those extended SFRs, the reader is referred to *[ICPP]*.

The following additional extended components are defined for this Security Target.

7.1 FMT_CMT Control over Management by TSF components

7.1.1 Family behavior

This family allows the specification of defined TSF components that take control over the management of TSF data.

The main difference between the existing components of Part 2 of *[CC]* and the one defined here is that the management function is not restricted to a specified role, but is restricted to defined TSF components.

This SFR does not have any dependencies on other management SFRs as it does not require administrative intervention since the management mechanism is hard-coded into the TSF.

7.1.2 Component leveling

FMT_CMT.1 Management of TSF data allows dedicated TSF components to manage TSF data.

7.1.3 Management: FMT_CMT.1

There are no management activities foreseen.

7.1.4 Audit: FMT_CMT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

Basic: All modifications to the values of TSF data.

7.1.5 FMT_CMT.1 management of TSF data by TSF components

Hierarchical to: No other components

Dependencies: No other components

FMT_CMT.1.1 The TSF shall restrict the ability to [selection: change_default, query, modify, delete, clear, [assignment: other operations]] the [assignment: list of TSF data] to [assignment: the defined TSF components].

7.2 FDP_SDA Stored Data Authenticity

7.2.1 Family behavior

This family provides requirements that address protection of user data and TSF data authenticity while these data are stored within memory areas protected by the TSF.

The TSF provides access to the data in the memory through the specified TOE interfaces only and detect modification of memory information bypassing these TOE interfaces.

The TSF complement the family Stored data integrity (FDP_SDI) which protects the user data from integrity errors while being stored in the memory.

7.2.2 Component leveling

FDP_SDA.1 Stored Data Authenticity requires the TOE to protect the authenticity of information of user data and TSF data in specified memory areas.

7.2.3 Management: FDP_SDA.1

There are no management activities foreseen.

7.2.4 Audit: FDP_SDA.1

There are no actions defined to be auditable

7.2.5 FDP_SDA.1 Stored Data Authenticity

Hierarchical to: No other components

Dependencies: No other components

FDP_SDA.1.1 The TSF shall ensure the authenticity of user data and TSF data while it is stored in the [assignment: memory area]

7.3 FDP_SDR Stored Data Replay protection

7.3.1 Family behavior

This family provides requirements that address protection of user data and TSF data against replay attack while these data are stored within memory areas protected by the TSF.

The TSF provides access to the data in the memory through the specified TOE interfaces only and detect replay of otherwise valid data bypassing these TOE interfaces.

It complement the family Stored data integrity (FDP_SDI) which protects the user data from integrity errors while being stored in the memory.

7.3.2 Component leveling

FDP_SDR.1 Stored Data Replay protection requires the TOE to protect against replay attack of information of user data and TSF data in specified memory areas.

7.3.3 Management: FDP_SDR.1

There are no management activities foreseen.

7.3.4 Audit: FDP_SDR.1

There are no actions defined to be auditable

7.3.5 FDP_SDR.1 Stored Data Replay Protection

Hierarchical to: No other components

Dependencies: No other components

FDP_SDR.1.1 The TSF shall detect replay of user data and TSF data while it is stored in the [assignment: memory area]

8 Security requirements

8.1 Security functional requirements

8.1.1 Security functional requirements from the body of the protection profile

FRU_FLT.2 – Limited fault tolerance

FRU_FLT.2.1 – The TSF shall ensure the operation of all the TOE's capabilities when the following failures occur: exposure to operating conditions that are not detected according to the requirement Failure with preservation of secure state (FPT_FLS.1).

Refinement: The term *failure* above means *circumstances*. The TOE prevents failures for the circumstances defined above.

FPT_FLS.1 – Failure with preservation of secure state

FPT_FLS.1.1 – The TSF shall preserve a secure state when the following types of failures occur: exposure to operating conditions which may not be tolerated according to the requirement Limited fault tolerance (FRU_FLT.2) and where therefore a malfunction could occur.

Refinement: The term *failure* above also covers *circumstances*. The TOE prevents failures for the circumstances defined above.

Application Note: The failures will cause alarm signals to be triggered which will result in an interrupt or a reset (secure state). This addresses the application note 14 in *[ICPP]*

FMT_LIM.1 – Limited capabilities

FMT_LIM.1.1 – The TSF shall be designed and implemented in a manner that limits their capabilities so that in conjunction with Limited availability (FMT_LIM.2) the following policy is enforced: Deploying Test Features after TOE Delivery does not allow user data of the Composite TOE to be disclosed or manipulated, TSF data to be disclosed or manipulated, software to be reconstructed, and no substantial information about construction of TSF to be gathered which may enable other attacks.

Application Note: *Composite TOE* refers to the Secure Processor hardware component part of the SoC (excluding the remainder of the SoC) with the firmware and software executing on the SP-CPU. Software executing on other parts of the SoC (including

software executing in the Arm TrustZone of the SoC master processor) is not considered here. Such software can be viewed similar to the software in entities external to the IC such as the software in a smart card reader. All such software in other parts of the SoC is considered as non-interfering.

FMT_LIM.2 – Limited availability

FMT_LIM.2.1 – The TSF shall be designed and implemented in a manner that limits their availability so that in conjunction with Limited capabilities (FMT_LIM.1) the following policy is enforced: Deploying Test Features after TOE Delivery does not allow user data of the Composite TOE to be disclosed or manipulated, TSF data to be disclosed or manipulated, software to be reconstructed, and no substantial information about construction of TSF to be gathered which may enable other attacks.

FAU_SAS.1 – Audit storage

FAU_SAS.1.1 – The TSF shall provide the test process before TOE Delivery with the capability to store [the Initialization Data, Pre-personalization Data] in the SP-QFPROM.

FDP_SDC.1(1) – Stored data confidentiality

FDP_SDC.1.1(1) – The TSF shall ensure the confidentiality of the information of the user data *and the TSF data* while it is stored in the memory area within TOE HW boundary.

Application Note: Different memory areas have different access protection mechanisms. Especially, key material stored in the SP-CMU key tables is confidentiality-protected even from any software executing on the SP-CPU. These keys can be either TSF data or user data and, therefore, the SFR has been refined to also include TSF data. The TSF data to which this applies are keys used by the TSF internally (hardware keys), while managed keys can be user data when they are defined for and used by an application executing on the operating system of the SP-CPU.

FDP_SDI.2(1) – Stored data integrity monitoring and action

FDP_SDI.2.1(1) – The TSF shall monitor user data stored in containers controlled by the TSF for parity errors using parity check/Error Correcting Code and errors after partial power collapse using a checksum function on all objects, based on the following attributes: data stored in SP-RAM, data stored in the SP-CMU key tables.

FDP_SDI.2.2(1) – Upon detection of a data integrity error, the TSF shall perform a cold reset and increment a fault counter.

Application Note: Another SFR, defined in section *8.1.3.2*, handles the special case of the operating system image stored in RAM.

FPT_PHP.3 – Resistance to physical attack

FPT_PHP.3.1 – The TSF shall resist physical manipulation and physical probing to the TSF by responding automatically such that the SFRs are always enforced.

Refinement from *[ICPP]*: The TSF will implement appropriate mechanisms to continuously counter physical manipulation and physical probing. Due to the nature of these attacks (especially manipulation), the TSF can by no means detect attacks on all of its elements. Therefore, permanent protection against these attacks is required ensuring that security functional requirements are enforced. Hence, automatic response means here (i) assuming that there might be an attack at any time and (ii) countermeasures are provided at any time.

FDP_ITT.1 – Basic internal transfer protection

FDP_ITT.1.1 – The TSF shall enforce the Data Processing Policy to prevent the [disclosure] of user data when it is transmitted between physically separated parts of the TOE.

Refinement: The different memories, the SP-CPU and other functional units of the TOE (for example a cryptographic co-processor) are seen as physically separated parts of the TOE.

FPT_ITT.1 – Basic internal TSF data transfer protection

FPT_ITT.1.1 – The TSF shall protect TSF data from [disclosure] when it is transmitted between separate parts of the TOE.

Refinement: The different memories, the CPU, and other functional units of the TOE (for example a cryptographic co-processor) are seen as physically separated parts of the TOE.

FDP_IFC.1 – Subset information flow control

FDP_IFC.1.1 – The TSF shall enforce the Data Processing Policy on all confidential data when they are processed or transferred by the TOE or by the Security IC Embedded Software.

The following Security Function Policy (SFP) Data Processing Policy is defined for the requirement Subset information flow control (FDP_IFC.1):

“User data of the Composite TOE and TSF data shall not be accessible from the TOE except when the Security IC Embedded Software decides to communicate the user data of the Composite TOE via an external interface. The protection shall be applied to confidential data only but without the distinction of attributes controlled by the Security IC Embedded Software.”

Application Note: Security IC Embedded Software in the case of the TOE is any user application running on the SP-CPU.

Application Note: The component FDP_IFC.1 in the *[CC]* has a dependency on FDP_IFF.1, which is not resolved in *[ICPP]*. The discussion of this omission in *[ICPP]* is not very convincing. Basically, this omission implies that the TOE has to decide which

data it considers to be confidential such that it shall not be exported over any of the TOE external interfaces.

FCS_RNG.1 – Random number generation (Class PTG.3)

FCS_RNG.1.1 – The TSF shall provide a [hybrid physical] random number generator that implements:

(PTG.3.1) A total failure test detects a total failure of entropy source immediately when the RNG has started. When a total failure has been detected, no random numbers will be output.

(PTG.3.2) If a total failure of the entropy source occurs while the RNG is being operated, the RNG [prevents the output of any internal random number that depends on some raw random numbers that have been generated after the total failure of the entropy source].

(PTG.3.3) The online test shall detect non-tolerable statistical defects of the raw random number sequence (i) immediately when the RNG is started, and (ii) while the RNG is being operated. The TSF must not output any random numbers before the power-up online test and the seeding of the DRG.3 post-processing algorithm have finished successfully or when a defect has been detected.

(PTG.3.4) The online test procedure shall be effective to detect non-tolerable weaknesses of the random numbers soon.

(PTG.3.5) The online test procedure checks the raw random number sequence. It is triggered [continuously]. The online test is suitable for detecting non-tolerable statistical defects of the statistical properties of the raw random numbers within an acceptable period of time.

(PTG.3.6) The algorithmic post-processing algorithm belongs to Class DRG.3 with cryptographic state transition function and cryptographic output function, and the output data rate of the post-processing algorithm shall not exceed its input data rate.

FCS_RNG.1.2 – The TSF shall provide [numbers in 32-bit blocks] that meet:

(PTG.3.7) Statistical test suites cannot practically distinguish the internal random numbers from output sequences of an ideal RNG. The internal random numbers must pass test procedure A [and procedure B].

(PTG.3.8) The internal random numbers shall [use PTRNG of class PTG.2 as random source for the post-processing].

8.1.2 Security functional requirements from augmentation packages

The following section contains security functional requirements from packages defined in *[ICPP]*:

- Cryptographic services package AES

- Cryptographic services package Hash functions

8.1.2.1 Cryptographic services package AES

This package is included to address the provision of Advanced Encryption Standard encryption/decryption.

The package's organizational security policy and security objectives were added to the respective lists in Sections *5* and *6* . This package adds the following SFRs.

8.1.2.1.1 Security functional requirements

FCS_COP.1/AES– Cryptographic operation – AES

FCS_COP.1.1/AES – The TSF shall perform decryption and encryption in accordance with a specified cryptographic algorithm AES in ECB mode, CBC mode, CTR mode, GCM mode (with additional authentication) and CCM mode (with additional authentication) and cryptographic key sizes [128 bit, 256 bit] that meet the following: FIPS 197, NIST SP 800-38A, NIST SP 800-38D and NIST SP 800-38C.

FCS_CKM.4/AES – Cryptographic key destruction – AES

FCS_CKM.4.1/AES – The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method overwriting with zeroes or overwriting the protecting key encryption key in the key hierarchy with ones that meets the following: none.

8.1.2.2 Cryptographic services package Hash functions

This package is included to address the provision of secure hash functions.

8.1.2.2.1 Security functional requirements

FCS_COP.1/SHA– Cryptographic operation – SHA

FCS_COP.1.1/SHA – The TSF shall perform hashing in accordance with a specified cryptographic algorithm [SHA-1, SHA-256, SHA-384, SHA-512] and cryptographic key sizes none that meet the following: Secure Hash Standard (SHS) (FIPS 180-4).

8.1.3 Security functional requirements beyond those in *[ICPP]*

The TOE implements a set of additional security functions that are beyond what is defined in *[ICPP]*. This includes functions provided by the hardware as well as functions provided by the operating system of the SP-CPU, which is part of the TOE and its TSF.

The SFRs for those functions are grouped for specific additional functions, which are described in general at the beginning of each group before stating the SFRs for those functions in CC terminology.

8.1.3.1 Group 1: Cryptographic functions

This group describes the cryptographic functions that are beyond those defined in *[ICPP]*. This includes the asymmetric algorithms.

FCS_COP.1/TDES – Cryptographic operation – TDES

FCS_COP.1.1/TDES – The TSF shall perform encryption and decryption in accordance with a specified cryptographic algorithm TDES in ECB mode, CBC mode and cryptographic key sizes [112 bit, 168 bit] that meet the following: NIST SP 800-67 and NIST SP 800-38A.

FCS_CKM.4/TDES Cryptographic key destruction – TDES

FCS_CKM.4.1/TDES – The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method overwriting with random values that meets the following: none.

FCS_CKM.1/SYM – Cryptographic key generation – Symmetric Keys

FCS_CKM.1.1/SYM – The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm capable of generating a random bit sequence and specified cryptographic key sizes :

- AES: 128 bits, 256 bits
- CMAC AES: 128 bits, 256 bits
- HMAC SHA-1: 256 bits
- HMAC SHA-256: 256 bits
- HMAC SHA-384: 256 bits
- HMAC SHA-512: 256 bits

that meet the following: NIST SP 800-133 Section 6 with V being a string of zeroes.

FCS_CKM.1/KDF – Cryptographic key generation – Key Derivation Function

FCS_CKM.1.1/KDF – The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm Key Derivation Function in Counter Mode using HMAC SHA-256 and specified cryptographic key sizes:

- AES: 128 bits, 256 bits
- CMAC AES: 128 bits, 256 bits
- HMAC SHA-1: 256 bits
- HMAC SHA-256: 256 bits
- HMAC SHA-384: 256 bits

- HMAC SHA-512: 256 bits

that meet the following: NIST SP 800-108 Section 5.1, FIPS 198-1, FIPS 180-4.

FCS_CKM.4/HMAC/CMAC – Cryptographic key destruction

FCS_CKM.4.1/HMAC/CMAC – The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method overwriting with zeroes for keys of size above 32 bytes or overwriting the protecting key encryption key in the key hierarchy with ones for all other keys that meets the following: none.

FCS_COP.1/CMAC – Cryptographic operation

FCS_COP.1.1/CMAC – The TSF shall perform message authentication code generation in accordance with a specified cryptographic algorithm CMAC using AES and cryptographic key sizes 128 bits, 256 bits that meet the following: FIPS 197 and NIST SP 800-38B.

FCS_COP.1/HMAC – Cryptographic operation

FCS_COP.1.1/HMAC – The TSF shall perform message authentication code generation in accordance with a specified cryptographic algorithm HMAC using SHA-1 or SHA-256, SHA-384, SHA-512 and cryptographic key sizes 256 bits for SHA-1 and SHA-256, 512 bits for SHA-384 and SHA-512 that meet the following: FIPS 198-1 and FIPS 180-4.

FCS_CKM.1/ECDH – Cryptographic key generation – ECDH

FCS_CKM.1.1/ECDH – The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm ECDH key generation with ECC key lengths corresponding to the ECC parameters for NIST P-192/224/256/384/521, Brainpool P-192/224/256/320/384/512 non-twisted (r1) and Curve25519 curves and specified cryptographic key sizes implied by the curve that meet the following: FIPS 186-4 Appendix B.4.2 supported by FIPS 186-4 Appendix D.1.2, RFC 5639 and RFC 7748.

FCS_CKM.1/ECDSA – Cryptographic key generation – ECDSA

FCS_CKM.1.1/ECDSA – The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm ECC key generation with ECC key lengths corresponding to the ECC parameters for NIST P-192/224/256/384/521 and Brainpool P-192/224/256/320/384/512 non-twisted (r1) curves and specified cryptographic key sizes implied by the curve that meet the following: FIPS 186-4 Appendix B.4.2 supported by FIPS 186-4 Appendix D.1.2 and RFC 5639.

FCS_CKM.1/RSA – Cryptographic key generation – RSA

FCS_CKM.1.1/RSA – The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm RSA key generation and specified cryptographic key sizes 1024 and 2048 bits that meet the following: FIPS 186-4 Appendix B.3 and BSI TR-02102-1.

FCS_CKM.4/RSA/ECDSA/ECDH – Cryptographic key destruction

FCS_CKM.4.1/RSA/ECDSA/ECDH– The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method overwriting with zeroes that meets the following: none.

FCS_COP.1/RSA_SIGN – Cryptographic operation

FCS_COP.1.1/RSA_SIGN – The TSF shall perform signature generation and verification in accordance with a specified cryptographic algorithm RSA and cryptographic key sizes 1024 bits, 2048 bits (RSA) that meet the following: FIPS 186-4 Section 5, PKCS#1 v2.1 PSS, PKCS#1 v1.5.

FCS_COP.1/RSA_ENC – Cryptographic operation

FCS_COP.1.1/RSA_ENC – The TSF shall perform encryption and decryption in accordance with a specified cryptographic algorithm RSA and cryptographic key sizes 1024 bits, 2048 bits (RSA) that meet the following: FIPS 186-4 Section 5, PKCS#1 v2.1 OAEP, PKCS#1 v1.5.

FCS_COP.1/ECDSA – Cryptographic operation

FCS_COP.1.1/ECDSA – The TSF shall perform signature generation and verification in accordance with a specified cryptographic algorithm ECDSA with ECC key lengths corresponding to the ECC parameters for NIST P-192/224/256/384/521 and Brainpool P-192/224/256/320/384/512 non-twisted (r1) curves and cryptographic key sizes implied by the curve that meet the following: FIPS 186-4 Section 6 supported by FIPS 186-4 Appendix D.1.2 and RFC 5639.

FCS_COP.1/ECDH – Cryptographic operation

FCS_COP.1.1/ECDH – The TSF shall perform shared secret generation in accordance with a specified cryptographic algorithm ECDH with ECC key lengths corresponding to the ECC parameters for NIST P-192/224/256/384/521, Brainpool P-192/224/256/320/384/512 non-twisted (r1) and Curve25519 curves and cryptographic key sizes implied by the curve that meet the following: NIST SP 800-56A Section 5.7.1.2 supported by FIPS 186-4 Appendix D.1.2, RFC 5639 and RFC 7748.

8.1.3.2 Group 2: Secure boot

This group describes the functions for the secure boot and startup process of the TOE. This includes the verification of the authenticity and integrity of the boot code and application executables, and the decryption of that data for a cold boot and the integrity verification of that data in SP-RAM in case of a warm boot. The case for the cold boot is defined using SFR FDP_ITC.1 and the case of the warm boot is defined using SFR FDP_SDI.2(2).

FDP_ITC.1 – Import of user data without security attributes

FDP_ITC.1.1 – The TSF shall enforce the no access control policy when importing user data, controlled under the SFP, from outside of the TOE.

FDP_ITC.1.2 – The TSF shall ignore any security attributes associated with the user data when imported from outside of the TOE.

FDP_ITC.1.3 – The TSF shall enforce the following rules when user data controlled under the SFP from outside the TOE:

- The TSF shall verify the digital signature of the boot image and application executables.
- The TSF shall verify the version number of the boot image and application executables to be equal or larger than the version number known to the TOE to prevent a rollback.
- The TSF shall stop execution when any of the aforementioned validation steps fail.
- When the aforementioned validation steps are successful, the TSF shall decrypt the boot image and application executables.

Application Note: Within this SFR, the TOE refers to SPU and the user data refers to boot image (MCP) and application executables.

FDP_SDI.2(2) – Stored data integrity monitoring and action

FDP_SDI.2.1(2) – The TSF shall monitor user data stored in containers controlled by the TSF for integrity errors on all objects, based on the following attributes: 64-bits SUM complemented value of the SP-RAM content.

FDP_SDI.2.2(2) – Upon detection of a data integrity error, the TSF shall start a cold boot process, which erases the SP-RAM.

Application Note: This SFR is a refinement for the specific user data operating system image and application executables and data stored in SP-RAM.

8.1.3.3 Group 3: Access control policies for hardware components

The TOE implements a set of policies that control access to critical hardware components, memory areas and keys. Access control to keys is also a TSF internal access control mechanism where access to keys from SP-CPU is controlled, in general, by the SP-CMU.

Those access control policies are related to TSF data rather than user data and are, therefore, expressed by extended SFRs for the management of TSF data.

FMT_CMT.1(1) – Management of TSF data by TSF components

FMT_CMT.1.1(1) – The TSF shall restrict the ability to [modify and define] the memory areas shared with other components of the SoC to SP-ExtMM controlled by the SP-CPU.

FMT_CMT.1(2) – Management of TSF data by TSF components

FMT_CMT.1.1(2) – The TSF shall restrict the ability to [modify and access] the key table to SP-CMU.

FMT_CMT.1(3) – Management of TSF data by TSF components

FMT_CMT.1.1(3) – The TSF shall restrict the ability to [use] the hardware support for cryptographic operations and the random number generator to SP-CMU.

Application Note: the term “use” means to access to RNG generated numbers or invoke operations support given by the hardware coprocessors. This term is not related to the access for configuration of the RNG or hardware.

FMT_CMT.1(4) – Management of TSF data by TSF components

FMT_CMT.1.1(4) – The TSF shall restrict the ability to [program] the SP-ExtMM and SP-MPU to SP-CPU when in privileged mode.

FMT_CMT.1(5) – Management of TSF data by TSF components

FMT_CMT.1.1(5) – The TSF shall restrict the ability to [access] the areas in the SP-ROM to SP-CPU based on the restrictions implemented by the SP-ROM permission checker.

FMT_CMT.1(6) – Management of TSF data by TSF components

FMT_CMT.1.1(6) – The TSF shall restrict the ability to [patch] the SP-ROM memory to the OTP patch logic and the CSR patch mechanism.

8.1.3.4 Group 4: Access control policies for software-managed data

The TOE implements a set of policies that control access to user and TSF data managed by the operating system. The operating system access control SFP relates to all processes managed by the TOE and relies on data encryption on a per process basis. Each process uses its own key and therefore prevents access to its data by other processes. A process can be the TOE operating system or an application running in the TOE.

FDP_ACC.2 – Complete access control

FDP_ACC.2.1 – The TSF shall enforce the operating system access control SFP on

- Subjects: operating system, applications, and
- Objects: all non-volatile data storage objects holding application data, application code, TSF data and TSF code

and all operations among subjects and objects covered by the SFP.

FDP_ACC.2.2 – The TSF shall ensure that all operations between any subject controlled by the TSF and any object controlled by the TSF are covered by an access control SFP.

FDP_ACF.1 – Security attribute based access control

FDP_ACF.1.1 – The TSF shall enforce the operating system access control SFP to objects based on the following:

- Subjects: all subjects are associated with dedicated encryption keys,
- Objects: non-volatile data storage objects encrypted with one of the encryption keys and associated anti-replay information.

FDP_ACF.1.2 – The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: non-volatile storage data will be encrypted with the subject-specific key during write operations and will be decrypted with that key during read operations, and will be marked with the anti-replay information to prevent data replay.

FDP_ACF.1.3 – The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: none.

FDP_ACF.1.4 – The TSF shall explicitly deny access of subjects to objects based on the following additional rules: none.

FMT_MSA.3 – Static attribute initialization

FMT_MSA.3.1– The TSF shall enforce the operating system access control SFP to provide [restrictive] default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2– The TSF shall allow nobody to specify alternative initial values to override the default values when an object or information is created.

FDP_RIP.1/Keys – Subset residual information protection

FDP_RIP.1.1/Keys – The TSF shall ensure that any previous information content of a resource is made unavailable upon the [deallocation of the resource from] the following objects: the interface buffers to the cryptographic services.

FDP_RIP.1/Transient - Subset residual information protection

FDP_RIP.1.1/Transient – The TSF shall ensure that any previous information content of a resource is made unavailable upon the [deallocation of the resource from] the following objects: any transient object.

8.1.3.5 Group 5: Protection of data stored outside the TOE

The TOE implements a set security mechanisms to protect user data and TSF dedicated data stored outside the TOE in memories shared with the host SoC.

The provided protection include integrity, confidentiality, authenticity and anti-replay to protect against eavesdropping, modification and replay attacks.

FDP_SDI.2(3) – Stored Data Integrity

FDP_SDI.2.1(3) – The TSF shall monitor user data stored in containers controlled by the TSF for any integrity errors on all objects, based on the following attributes: SHA-256 hash tree and CMAC-AES-256 for persistent data store and AES-CCM-256 for volatile data store *while data is stored in the memory area outside the TOE hardware boundary.*

FDP_SDI.2.2(3) – Upon detection of a data integrity error, the TSF shall prevent the data from being used by the TOE.

FDP_SDC.1(2) – Stored Data Confidentiality

FDP_SDC.1.1(2) – The TSF shall ensure the confidentiality of the information of the user data *and the TSF data* while it is stored in the memory area outside the TOE hardware boundary.

Application Note: The TOE encrypts (AES-CBC-256) persistent user data before being exported outside the TOE (NVM storage use case). Confidentiality is ensured by the fact that the TOE generates a cryptographic key using a random number to encrypt the user data. This key is not known outside of the TOE nor is it exportable.

Application Note: The TOE encrypts (AES-CCM-256) transient user data before being exported outside the TOE (swap use case). Confidentiality is ensured by the fact that the TOE generates a cryptographic key using a random number to encrypt the user data. This key is not known outside of the TOE nor is it exportable.

FDP_SDA.1 – Stored Data Authenticity

FDP_SDA.1.1 – The TSF shall ensure the authenticity of the information of user data and TSF data while it is stored in the memory area outside the TOE hardware boundary.

Application Note: The TOE is capable of identifying whether user data and TSF data originate from the TOE or not. Only such data is loaded back into the TOE for processing. This is ensured by the fact that the user data is always authenticated by using a dedicated cryptographic key, generated within TOE. This key is generated using a random number and is not known outside of the TOE nor is it exportable.

Application Note: For persistent data, the key is persistent across reset. For transient data, the key is volatile (stored in SP-RAM).

FDP_SDR.1 – Stored Data Replay Protection

FDP_SDR.1.1 – The TSF shall detect replay of user data and TSF data while it is stored in the memory area outside the TOE hardware boundary.

Application Note: The TOE is capable of identifying whether user data is replay or not (current version of user data is replaced by an older version with valid encryption / authentication code by an attacker). This is ensured by the fact that the TOE manages and stores internally a monotonic counter that is used in the computation of the authentication code as defined in FDP_SDI.2(3).

Application Note: For persistent data, the counter is managed by the Anti-Replay Island (ARI) and is persistent across reset. For transient data, the counter is volatile (stored in SP-RAM).

8.2 Security assurance requirements

The Security Assurance Requirements are as defined in *[ICPP]*, including the refinements defined there.

8.3 Security requirements rationale

Section 6.3 of *[ICPP]* defines the security requirements rationale for the objectives and SFRs defined in *[ICPP]*, and Section 7 of *[ICPP]* defines the additional security requirements rationale for the optional packages. This rationale applies for all security objectives and SFRs taken from *[ICPP]*. This section, therefore, addresses only the rationale for the additional security objectives and SFRs defined in this Security Target and for the additional objectives and SFRs from the optional packages “AES” and “Hash functions” from *[ICPP]*.

This Security Target adds the following additional security objectives:

- O.Defense-in-Depth
- O.Secure-Boot
- O.SW-TDES
- O.RSA
- O.ECDSA
- O.ECDH
- O.KDF
- O.CMAC
- O.HMAC
- O.OSData.Access
- O.OSData.Protect
- O.Reallocation

Those security objectives are addressed by the following additional SFRs. In addition, the following table includes the objectives rationale for the selected optional packages defined in *[ICPP]*.

Table 8-1 Security Requirement vs Objectives mapping

Objective	Description
O.AES	The security objective of the TOE to provide secure hardware based cryptographic services for the AES for encryption and decryption is addressed by the SFRs of FCS_COP.1/AES, FCS_CKM.1/SYM and FCS_CKM.4/AES defining the cryptographic operation of the cipher and the associated functionality of key generation and destruction.
O.SHA	The security objective of the TOE to provide secure hardware-based cryptographic services for secure hash calculation is addressed by the SFR of FCS_COP.1/SHA defining the cryptographic operation of the cipher.
O.CMAC	The security objective of the TOE to provide secure hardware based cryptographic services for the CMAC-AES is addressed by the SFRs of FCS_COP.1/CMAC and FCS_CKM.4/HMAC/CMAC and FCS_CKM.1/SYM defining the cryptographic operation of the cipher used for Message Authentication Code generation and the associated functionality of key generation and destruction. CMAC keys larger than 32-byte are not protected by the SP-CMU (key table mechanism).
O.HMAC	The security objective of the TOE to provide secure hardware based cryptographic services for the HMAC is addressed by the SFRs of FCS_COP.1/HMAC and FCS_CKM.4/HMAC/CMAC and FCS_CKM.1/SYM defining the cryptographic operation of the cipher used for Message Authentication Code generation and the associated functionality of key destruction. HMAC keys larger than 32-byte are not protected by the SP-CMU (key table mechanism).
O.Defense-in-Depth	The security objective of the TOE to ensure that critical functions and TSF data cannot be accessed by a simple breach of security of the software executing on the SP-CPU is addressed by the SFRs FMT_CMT.1(1) to FMT_CMT.1(6) which define various security functions offered by the SPU that can and must be utilized by the TOE.
O.Secure-Boot	The security objective of the TOE to ensure that only authorized software is loaded during the boot process after the integrity and authenticity of that software has been verified is addressed by the SFRs FDP_ITC.1 and FDP_SDI.2(2). Both define the integrity protection mechanism of the software imported from outside of the TOE.
O.SW-TDES	The security objective of the TOE to provide secure software based cryptographic services for the TDES for encryption and decryption is addressed by the SFRs of FCS_COP.1/TDES and FCS_CKM.4/TDES defining the cryptographic operation of the cipher and the associated functionality of key destruction.
O.RSA	The security objective of the TOE to provide secure cryptographic services implementing the RSA algorithm for signature generation and verification and encryption and decryption is addressed by the SFRs FCS_CKM.1/RSA, FCS_CKM.4/RSA/ECDSA/ECDH, FCS_COP.1/RSA_SIGN and FCS_COP.1/RSA_ENC which collectively define the key generation, destruction and cryptographic operation of RSA.
O.ECDSA	The security objective of the TOE to provide secure cryptographic services implementing the ECDSA algorithm based on the NIST P-192/224/256/384/521 and Brainpool P-192/224/256/320/384/512 non-twisted (r1) curves for signature generation and verification is addressed by the SFRs FCS_CKM.1/ECDSA, FCS_CKM.4/RSA/ECDSA/ECDH and FCS_COP.1/ECDSA which collectively define the key generation, destruction and cryptographic operation of ECDSA.

Objective	Description
O.ECDH	The security objective of the TOE to provide secure cryptographic services implementing the ECDH algorithm based on the NIST P-192/224/256/384/521, Brainpool P-192/224/256/320/384/512 non-twisted (r1) and Curve25519 curves for shared secret generation is addressed by the SFRs FCS_CKM.1/ECDH, FCS_CKM.4/RSA/ECDSA/ECDH, and FCS_COP.1/ECDH which collectively define the key generation, destruction and cryptographic operation of ECDH.
O.KDF	The security objective of the TOE to provide secure cryptographic services implementing the Key Derivation Function algorithm based on NIST SP 800-108 is addressed by the SFR FCS_CKM.1/KDF specifying such key derivation function.
O.OSData.Access	The security objective of the TSF to restrict access of software applications exclusively to its own data stored in volatile and non-volatile memory is addressed by the SFRs of FDP_ACC.2, FDP_ACF.1, FMT_MSA.3 which collectively define such access control mechanism.
O.OSData.Protect	The security objective of the TSF to protect TSF code, TSF data, application code and application data using cryptographic functions with a cryptographic strength commensurate with the value of the data against loss of confidentiality, integrity, authenticity and against replay is addressed by the SFRs of FDP_ACC.2, FDP_ACF.1, FMT_MSA.3, FDP_SDI.2(3), FDP_SDI.2(2), FDP_SDC.1(2), FDP_SDA.1, FDP_SDR.1, FDP_RIP.1/Keys and FDP_RIP.1/Transient, which collectively define such protection control mechanism. The Hardware managed symmetric keys maintained by the TOE are protected as specified in FMT_CMT.1(2).
O.Reallocation	The security objective of the TOE to ensure that the re-allocation of a memory block for the runtime areas maintained by the TOE operating system does not disclose any information that was previously stored is addressed by the SFRs of FDP_RIP.1/Keys, FDP_RIP.1/Transient which define a clearing of residual information from storage locations upon the deallocation of the resource.

The following table provides the SFR dependency analysis for the SFRs covering the aforementioned objectives.

Table 8-2 Security Requirement dependencies

SFR	Dependencies	Fulfillment
FCS_CKM.1/SYM	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4	FCS_COP.1/AES FCS_COP.1/CMAC FCS_COP.1/HMAC FCS_CKM.4/AES FCS_CKM.4/HMAC/CMAC
FCS_CKM.1/KDF	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4	FCS_COP.1/AES FCS_COP.1/CMAC FCS_COP.1/HMAC FCS_CKM.4/AES FCS_CKM.4/HMAC/CMAC
FCS_CKM.1/ECDH	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4	FCS_COP.1/ECDH FCS_CKM.4/RSA/ECDSA/ECDH
FCS_CKM.1/ECDSA	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4	FCS_COP.1/ECDSA FCS_CKM.4/RSA/ECDSA/ECDH

SFR	Dependencies	Fulfillment
FCS_CKM.1/RSA	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4	FCS_COP.1/RSA_SIGN FCS_COP.1/RSA_ENC FCS_CKM.4/RSA/ECDSA/ECDH
FCS_CKM.4/AES	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1]	FCS_CKM.1/SYM
FCS_CKM.4/TDES	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1]	FCS_CKM.1/TDES is not fulfilled. Indeed TDES implementation does not leverage the CMU HW to protect the TDES key, so TDES key management is left to the caller.
FCS_CKM.4/HMAC/CMAC	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1]	FCS_CKM.1/SYM FCS_CKM.1/KDF
FCS_CKM.4/RSA/ECDSA/ ECDH	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1]	FCS_CKM.1/ECDH FCS_CKM.1/ECDSA FCS_CKM.1/RSA
FCS_COP.1/AES	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/SYM FCS_CKM.1/KDF FCS_CKM.4/AES
FCS_COP.1/TDES	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/TDES is not fulfilled. Indeed TDES implementation does not leverage the CMU HW to protect the TDES key, so TDES key management is left to the caller. FCS_CKM.4/TDES
FCS_COP.1/SHA	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	The hash generation does not require any key which implies that the dependency on key generation or key import is not applicable and thus unmet. The hash generation does not use cryptographically sensitive parameters which implies that no such parameters must be destroyed. Thus, the dependency on key destruction is unmet.
FCS_COP.1/CMAC	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/SYM FCS_CKM.1/KDF FCS_CKM.4/HMAC/CMAC
FCS_COP.1/HMAC	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/SYM FCS_CKM.1/KDF FCS_CKM.4/HMAC/CMAC
FCS_COP.1/ECDSA	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/ECDSA FCS_CKM.4/RSA/ECDSA/ECDH
FCS_COP.1/ECDH	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/ECDH FCS_CKM.4/RSA/ECDSA/ECDH
FCS_COP.1/RSA_SIGN	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/RSA FCS_CKM.4/RSA/ECDSA/ECDH

SFR	Dependencies	Fulfillment
FCS_COP.1/RSA_ENC	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/RSA FCS_CKM.4/RSA/ECDSA/ECDH
FDP_ITC.1	[FDP_ACC.1 or FDP_IFC.1] FMT_MSA.3	The data forming the TOE software and that is imported by the TOE is encrypted and signed. No access control policy or information flow control is required because the associated keys are stored inside the TOE. By relying on the cryptographic protection the TOE ensures the application of the rules defined FDP_ITC.1.
FDP_SDI.2(2)	No dependencies	
FMT_CMT.1(1)	No dependencies	
FMT_CMT.1(2)	No dependencies	
FMT_CMT.1(3)	No dependencies	
FMT_CMT.1(4)	No dependencies	
FMT_CMT.1(5)	No dependencies	
FMT_CMT.1(6)	No dependencies	
FDP_ACC.2	FDP_ACF.1	FDP_ACF.1
FDP_ACF.1	FDP_ACC.1 FMT_MSA.3	FDP_ACC.2 FMT_MSA.3
FMT_MSA.3	FMT_MSA.1 FMT_SMR.1	The dependency of FMT_MSA.3 to FMT_MSA.1 and FMT_SMR.1 is unmet, because the default value cannot be altered or managed.
FDP_RIP.1/Keys	No dependencies	
FDP_RIP.1/Transient	No dependencies	
FDP_SDI.2(3)	No dependencies	
FDP_SDC.1(2)	No dependencies	
FDP_SDA.1	No dependencies	
FDP_SDR.1	No dependencies	

NOTE: The security objectives of O.Leak-Inherent, O.Phys-Probing, O.Malfunction, O.Phys-Manipulation, and O.Leak-Forced defined in *[ICPP]* support the secure implementation of all crypto services introduced by P.Crypto-Service and the P.SW_Crypto-Service. The TOE will ensure the confidentiality of the user data and TSF data for these crypto services.

9 TOE summary specification

This section describes the implementation of the security functional requirements by the TOE. It is important to note, that the most security functions in this sections are implemented by the SPU subsystem.

The TME subsystem participates in the secure boot of the entire platform and in particular, the only activities in the scope of the evaluation as SFR-supporting are the related to the boot of the SPU, related only to sections 8.1.3.2 and 9.1.2.1.

9.1 TOE summary specification rationale

The following table maps the SFRs to the sections of the TSS containing the descriptions of how those SFRs are implemented.

Table 9-1 TOE summary specification rationale

SFR	9.1.1.1 Random number generation	9.1.1.2 AES coprocessor	9.1.1.3 Hashing	9.1.1.4 Authentication	9.1.1.5 Key Derivation Function	9.1.1.6 Key protection	9.1.1.7 TDES	9.1.1.8 RSA, ECDSA, ECDH	9.1.2.1 Secure boot	9.1.2.2 Secure software update	9.1.3 Application manager	9.1.4 Domain separation between applications executed by the TOE	9.1.5 Physical protection	9.1.6.1 Control memory areas shared with other components of the SoC	9.1.6.2 Control access to keys and the key table	9.1.6.3 Control use of hardware support for cryptographic operations and random umber generation	9.1.6.4 Control access to the SP-RAM by software on the SP-CPU	9.1.7.1 Cryptographic protection of persistent data stored outside the TOE	9.1.7.2 Cryptographic protection of transient data and code stored outside the TOE	9.1.7.3 Reallocation of shared resources	9.1.8.1 Logical protection, SP-ROM	9.1.8.2 Logical protection, SP-RAM	9.1.9 Production data and OTP handling	9.1.10 Life-Cycle Control
FRU_FLT.2													X											
FPT_FLS.1													X											
FMT_LIM.1																								X
FMT_LIM.2																								X
FAU_SAS.1																							X	
FDP_SDC.1(1)																					X			
FDP_SDI.2(1)						X															X			
FPT_PHP.3													X											
FDP_ITT.1													X											
FPT_ITT.1													X											
FDP_IFC.1													X											
FCS_RNG.1	X																							
FCS_COP.1/AES		X																						
FCS_CKM.4/AES						X																		
FCS_COP.1/TDES							X																	
FCS_CKM.4/TDES							X																	

SFR	9.1.1.1 Random number generation	9.1.1.2 AES coprocessor	9.1.1.3 Hashing	9.1.1.4 Authentication	9.1.1.5 Key Derivation Function	9.1.1.6 Key protection	9.1.1.7 TDES	9.1.1.8 RSA, ECDSA, ECDH	9.1.2.1 Secure boot	9.1.2.2 Secure software update	9.1.3 Application manager	9.1.4 Domain separation between applications executed by the TOE	9.1.5 Physical protection	9.1.6.1 Control memory areas shared with other components of the SoC	9.1.6.2 Control access to keys and the key table	9.1.6.3 Control use of hardware support for cryptographic operations and random umber generation	9.1.6.4 Control access to the SP-RAM by software on the SP-CPU	9.1.7.1 Cryptographic protection of persistent data stored outside the TOE	9.1.7.2 Cryptographic protection of transient data and code stored outside the TOE	9.1.7.3 Reallocation of shared resources	9.1.8.1 Logical protection, SP-ROM	9.1.8.2 Logical protection, SP-RAM	9.1.9 Production data and OTP handling	9.1.10 Life-Cycle Control
FCS_COP.1/RSA_SIGN								X																
FCS_COP.1/RSA_ENC								X																
FCS_CKM.1/RSA								X																
FCS_COP.1/ECDSA								X																
FCS_CKM.1/ECDSA								X																
FCS_COP.1/ECDH								X																
FCS_CKM.1/ECDH								X																
FCS_CKM.4/RSA/ECDSA/ECDH						X																		
FCS_COP.1/SHA			X																					
FCS_CKM.1/SYM	X																							
FCS_CKM.1/KDF					X																			
FCS_CKM.4/HMAC/CMAC						X																		
FCS_COP.1/CMAC				X																				
FCS_COP.1/HMAC				X																				
FDP_ITC.1									X	X	X													
FDP_SDI.2(2)						X																	X	
FMT_CMT.1(1)														X										
FMT_CMT.1(2)															X									

SFR	9.1.1.1 Random number generation	9.1.1.2 AES coprocessor	9.1.1.3 Hashing	9.1.1.4 Authentication	9.1.1.5 Key Derivation Function	9.1.1.6 Key protection	9.1.1.7 TDES	9.1.1.8 RSA, ECDSA, ECDH	9.1.2.1 Secure boot	9.1.2.2 Secure software update	9.1.3 Application manager	9.1.4 Domain separation between applications executed by the TOE	9.1.5 Physical protection	9.1.6.1 Control memory areas shared with other components of the SoC	9.1.6.2 Control access to keys and the key table	9.1.6.3 Control use of hardware support for cryptographic operations and random umber generation	9.1.6.4 Control access to the SP-RAM by software on the SP-CPU	9.1.7.1 Cryptographic protection of persistent data stored outside the TOE	9.1.7.2 Cryptographic protection of transient data and code stored outside the TOE	9.1.7.3 Reallocation of shared resources	9.1.8.1 Logical protection, SP-ROM	9.1.8.2 Logical protection, SP-RAM	9.1.9 Production data and OTP handling	9.1.10 Life-Cycle Control
FMT_CMT.1(3)																X								
FMT_CMT.1(4)												X					X							
FMT_CMT.1(5)																					X			
FMT_CMT.1(6)																					X			
FDP_ACC.2											X													
FDP_ACF.1											X													
FMT_MSA.3											X													
FDP_RIP.1/Keys						X																		
FDP_RIP.1/Transient																				X				
FDP_SDI.2(3)																		X	X					
FDP_SDC.1(2)																		X	X					
FDP_SDA.1																		X	X					
FDP_SDR.1																		X	X					

9.1.1 Cryptographic services and random number generation

9.1.1.1 Random number generation

The implemented hybrid physical random number generator together with the associated post processing provide the functionality defined by FCS_RNG.1 and FCS_CKM.1/SYM.

9.1.1.2 AES coprocessor

A first AES coprocessor is implemented as part of the SP-CMU. The AES coprocessor with support of CryptoLib, which is implemented as part of the IC dedicated software, provides AES encryption and decryption with the various crypto modes as required by FCS_COP.1/AES.

9.1.1.3 Hashing

A coprocessor implemented as part of the SP-CMU supports the different hashing algorithms as required by FCS_COP.1/SHA. The coprocessor needs to be configured with the required hashing algorithm before the operation is started.

9.1.1.4 Authentication

A second AES coprocessor implemented as part of the SP-CMU provides AES authentication functionality as required by FCS_COP.1/CMAC.

A SHA co-processor supports an HMAC implementation as required by FCS_COP.1/HMAC.

9.1.1.5 Key Derivation Function

The SP-CMU subsystem provides the key derivation functionality as required by FCS_CKM.1/KDF.

9.1.1.6 Key protection

The key table evaluates the key properties and ensures that keys are only used for the intended usage. In addition, the implementation of the key table limits access to the keys by SP-CPU. Keys released by the software cannot be further used. Thereby, the key table implements the protection of keys as required by FCS_CKM.4/AES, FCS_CKM.4/HMAC/CMAC and FDP_RIP.1/Keys.

For RSA/ECDH/ECDSA, once the key is released, it is destructed by as required by FCS_CKM.4/RSA/ECDSA/ECDH.

The key table is also protected through the implementation of parity control as per FDP_SDI.2(1).

9.1.1.7 TDES

The TDES is implemented in the software of the TOE and provides TDES encryption and decryption with the various crypto modes as required by FCS_COP.1/TDES and key destruction as required by FCS_CKM.4/TDES.

9.1.1.8 RSA, ECDSA, ECDH

An asymmetric cryptography coprocessor implemented in hardware supplemented by a cryptographic library implemented in the TOE provides RSA, ECDSA and ECDH cryptographic operations as required by FCS_COP.1/RSA_SIGN, FCS_COP.1/RSA_ENC, FCS_COP.1/ECDSA and FCS_COP.1/ECDH as well as key generation as required by FCS_CKM.1/RSA, FCS_CKM.1/ECDSA, FCS_CKM.1/ECDH.

9.1.2 Secure boot and secure update

9.1.2.1 Secure boot

The MCP is stored outside the TOE and loaded during startup by the PBL stored in the SP-ROM as part of the TOE. After being enabled by the TME, the PBL performs the following cold boot sequence:

1. At power on, the TME Sequencer and APPS PBL are booted up in parallel.

2. The APPS PBL loads the TME Sequencer Software and the TME Core Software in TME Sequencer RAM and TME CPU RAM, respectively.

3. The TME Sequencer Firmware in ROM authenticates the TME Sequencer Software image in TME Sequencer RAM.

4. TME PBL authenticates the TME Core Software image in TME CPU RAM and jumps to it through the TME Mission ROM.

5. Beyond this point, APPS PBL is in charge of loading sequentially the different components of the entire platform and TME Core Software in charge of authenticating them and taking them out of reset in case of successful authentication.

6. When is time to boot up the SPU, the hypervisor checks the status of the SP-CPU and sends this status information to TME Core Software in RAM through a dedicated channel.

7. As the TME Core Software in RAM is the only component with the ability to bring the SP-CPU out of reset, it proceeds by polling a secure register to verify the status of the secure processor and in case positive, brings the SP-CPU out of reset.

8. SP-PBL initializes all memory areas (programming of the SP-MPU of the SP-CPU).

9. SP-PBL copies the MCP image from the DDR to the SP-RAM (not accessible by the rest of the SoC).

10. SP-PBL verifies the digital signature (ECC P-384 and SHA-256) of the (encrypted) MCP image stored in the SP-RAM. If this verification fails, the execution stops.

11. SP-PBL decrypts the MCP image.

12. SP-PBL passes control to the MCP by jumping to the MCP.

The MCP does not maintain specific properties or access control during the storage outside the TOE. The protection relies on the digital signature, the encryption and the protected version number as required by FDP_ITC.1.

9.1.2.2 Secure software update

In case the SPU IC dedicated software detects an updated MCP during the verification process, the new software version is verified in the same way. In addition, the version number maintained in the SP-QFPROM is verified and updated accordingly in case a greater version number is identified. Thereby the update functionality implements the security mechanisms required by FDP_ITC.1.

9.1.3 Application manager

The application manager uses the same security mechanisms required by FDP_ITC.1. This comprises the integrity and authenticity verification based on a valid signature, the decryption of the application image and the rollback protection for version control. User and TSF data belonging to a dedicated application is encrypted on process basis with different keys to enforce the access control policy required by FDP_ACC.2, FDP_ACF.1 and FMT_MSA.3. Thereby the user data and the TSF data of different applications is only accessible by the associated application.

9.1.4 Domain separation between applications executed by the TOE

The TOE implements a dedicated control for the access to external memory as well as for the access to internal memories and resources that can be configured in privileged mode. This separation is required by FMT_CMT.1(4).

9.1.5 Physical protection

The TOE provides protection mechanisms against non-invasive, semi-invasive, and invasive physical attacks. These countermeasures protect against side-channel attacks, fault injection attacks, and against the various physical attacks.

The TOE implements countermeasures against side-channel attacks including constant execution time, masking (for example by the AES coprocessors) and random polarity switching for the SP-CPU bus as required by FDP_ITT.1, FPT_ITT.1 and FDP_IFC.1.1.

The TOE implements countermeasures against fault injection attacks comprising redundant logic and error detection/correction code for the following components SP-CPU, SP-QFPROM, SP-ROM, SP-RAM, SP-KT and TOE internal buses as required by FPT_FLS.1.

Further on, the TOE includes sensors to detect invalid operational conditions. Sensors are implemented to control the TOE internal voltages, the TOE internal frequency and the TOE internal temperature. In addition, the TOE implements parity checks, redundancy checks and fault detection logic as required by FPT_FLS.1.

Internal clock generation and filtering of the power supply provides the limited fault tolerance as required by FRU_FLT.2.

General countermeasures like the generation of additional temporal noise by various means (software and hardware), the memory protection mechanism implemented for SP-ROM and SP-RAM,SP-QFPROM as well as the RTL obfuscation together with specific options and constraints for the physical layout provide the protection as required by FPT_PHP.3.

9.1.6 Access control and management (hardware)

9.1.6.1 Control memory areas shared with other components of the SoC

Access to the external RAM used to share data with other components of the SoC is controlled by a dedicated memory manager as required by FMT_CMT.1(1). The memory manager is able to support different windows in parallel.

9.1.6.2 Control access to keys and the key table

Access to keys stored in the key table is limited to the SP-CMU and does not allow the IC dedicated software executed on the SP-CPU or other external components to compromise keys via the software. This protection is required by FMT_CMT.1(2).

9.1.6.3 Control use of hardware support for cryptographic operations and random number generation

The access to coprocessors and the random number generator providing the cryptographic support to the IC dedicated software is limited to SP-CMU. This protection is required by FMT_CMT.1(3).

9.1.6.4 Control access to the SP-RAM by IC dedicated software on the SP-CPU

Access to the SP-RAM of the TOE is controlled by SP-MPU that can only be configured in privileged SP-CPU mode as required by FMT_CMT.1(4).

9.1.7 Access control and management (operating system)

9.1.7.1 Cryptographic protection of persistent data stored outside the TOE

The confidentiality, integrity, authenticity and replay-protection of data stored in the non-volatile memory outside the TOE boundary is ensured by cryptographic mechanisms. The encryption of the data is required by FDP_SDC.1(2), the integrity protection is required by FDP_SDI.2(3), the authenticity is required by FDP_SDA.1 and the replay protection is required by FDP_SDR.1.

9.1.7.2 Cryptographic protection of transient data and code stored outside the TOE

The confidentiality, integrity, authenticity and replay-protection of data stored in the DDR memory outside the TOE boundary is ensured by cryptographic mechanisms. The encryption of the data is required by FDP_SDC.1(2), the integrity protection is required by FDP_SDI.2(3), the authenticity is required by FDP_SDA.1 and the replay protection is required by FDP_SDR.1.

9.1.7.3 Reallocation of shared resources

The operating system implements the protection of memory areas that may get accessible by other applications or processes based on re-allocation of resources. Buffers with sensitive parameters and memory areas that are de-allocated are cleared as required by FDP_RIP.1/Transient.

9.1.8 Logical protection

9.1.8.1 SP-ROM

The SP-ROM is only accessible to the SP-CPU based on the control of the SP-MPU and the associated permission checker of SP-ROM as required by FMT_CMT.1(5). The ROM is partitioned and ROM content can only be patched by OTP patch (SP-QFPROM) logic or CSR patch mechanism as required by FMT_CMT.1(6). The SP-ROM implements memory scrambling and parity control as required by FDP_SDC.1(1) .

9.1.8.2 SP-RAM

The SP-RAM access is restricted to SP-CPU based on the control of the SP-MPU and the SP-DMA controller as well as the associated permission checker of SP-RAM. The SP-RAM implements memory encryption and parity control as required by FDP_SDC.1(1) and FDP_SDI.2(1). Further on a specific integrity check during warm boot is implemented as required by FDP_SDI.2(2).

9.1.9 Production data and OTP handling

Data for the associated initialization and pre-personalization data is stored in the SP-QFPROM as required by FAU_SAS.1.

9.1.10 Life-Cycle Control

The TOE implements Life-Cycle control based on a combination of access control to TOE functionality and the coding of the Life-Cycle phase in the SP-QFPROM. This implements the requirements FMT_LIM.1 and FMT_LIM.2.

A Cryptographic mechanisms table

Purpose	Cryptographic Mechanism	Standard of Implementation	Key Size in Bits / Key types	Key Origin
Image Authenticity and Integrity	RSA-signature verification with SHA-256 and PKCS#1 1.5 padding	FIPS 186-4 FIPS 180-4 RFC 3447	2048	Qualcomm managed private key
Image Confidentiality	AES-CBC	FIPS 197 NIST SP 800-38A	128	Qualcomm managed symmetric key
User and TOE Data while in NVM Confidentiality	AES-CBC	FIPS 197 NIST SP 800-38A	256	Key generated and managed by TOE operating system using SP-RNG
User and TOE Data while in NVM Integrity & Authenticity	SHA-256 tree and AES in CMAC mode	FIPS 180-4 FIPS 197 NIST SP 800-38B Hash tree specifics provided in developer document	256	Key generated and managed by TOE operating system using SP-RNG
User and TOE Data while in external DDR Confidentiality	AES-CBC	FIPS 197 NIST SP 800-38A	256	Key generated and managed by TOE operating system using SP-RNG
User and TOE Data while in external DDR Integrity & Authenticity	SHA-256	FIPS 180-4	256	N/A Reference hash stays in TOE
API	AES (ECB, CBC, CTR, CMAC, GCM, CCM)	FIPS 197 NIST SP 800-38A NIST SP 800-38B NIST SP 800-38C NIST SP 800-38D NIST SP 800-38E	128, 256	User-provided key managed by the TOE or key generated and managed by TOE operating system or SP-CMU using SP-RNG
API	SHA1 SHA-256 SHA-384 SHA-512	FIPS 180-4	none	N/A
API	RNG	PTG.3 of BSI-AIS20/AIS31 NIST SP 800-90A	none	N/A

Purpose	Cryptographic Mechanism	Standard of Implementation	Key Size in Bits / Key types	Key Origin
API	HMAC-SHA1 HMAC-SHA256 HMAC-SHA-384 HMAC-SHA-512	FIPS 198-1	256 (SHA-1, SHA-256) 512 (SHA-384, SHA-512)	User-provided key managed by the TOE or key generated and managed by TOE operating system or SP-CMU using SP-RNG
API	Generate random symmetric key	NIST SP 800-133	128, 192, 256	
API	Key Derivation Function in Counter Mode based on HMAC SHA-256	NIST SP 800-108 FIPS 198-1 FIPS 180-4	256	
API	TDES (ECB, CBC)	FIPS 46-3 NIST SP 800-38A NIST SP 800-67	112,168	User-provided key managed by the TOE operating system
API	RSA signature/verify with SHA-1, SHA-256, SHA-384, SHA-512 and RSASSA-PSS and PKCS#1 v1.5 padding schemes	FIPS 186-4 FIPS 180-4 RFC 3447	1024, 2048	User-provided key or key generated by TOE operating system using SP-RNG
API	RSA encryption / decryption with RSAES-OAEP and PKCS#1 v1.5 padding schemes	FIPS 186-4 RFC 3447	1024, 2048	User-provided key or key generated by TOE operating system using SP-RNG
API	ECDSA Cryptographic key generation Signature Generation / Verification	FIPS 186-4 RFC 5639 RFC 7748	ECC key lengths corresponding to following ECC parameters: NIST P-192 NIST P-224 NIST P-256 NIST P-384 NIST P-521 Brainpool P-192 (r1) Brainpool P-224 (r1) Brainpool P-256 (r1) Brainpool P-320 (r1) Brainpool P-384 (r1) Brainpool P-512 (r1)	User-provided key or key generated by TOE operating system (MCP) using SP-RNG

Purpose	Cryptographic Mechanism	Standard of Implementation	Key Size in Bits / Key types	Key Origin
API	ECDH Cryptographic key generation Shared secret generation	NIST SP 800-56A FIPS 186-4 RFC 5639 RFC 7748	ECC key lengths corresponding to following ECC parameters: NIST P-192 NIST P-224 NIST P-256 NIST P-384 NIST P-521 Brainpool P-192 (r1) Brainpool P-224 (r1) Brainpool P-256 (r1) Brainpool P-320 (r1) Brainpool P-384 (r1) Brainpool P-512 (r1) Curve25519	User-provided key or key generated by TOE operating system (MCP) using SP-RNG
API	RSA key generation	FIPS 186-4 BSI TR-02102-1	1024, 2048	Seed comes from SP-RNG

B References

B.1 Related documents

Title	Number
Standards	
Common Criteria for Information Technology Security Evaluation, Parts 1, 2, and 3, Version 3.1, Revision 5; [CC]	Part 1: CCMB-2017-04-001 Part 2: CCMB-2017-04-002 Part 3: CCMB-2017-04-003
DATA ENCRYPTION STANDARD (DES), October 1999	FIPS 46-3
Secure Hash Standard (SHS), August 2015	FIPS 180-4
Digital Signature Standard (DSS), July 2013	FIPS 186-4
ADVANCED ENCRYPTION STANDARD (AES), November 2001	FIPS 197
The Keyed-Hash Message Authentication Code (HMAC), July 2008	FIPS 198-1
Recommendation for Block Cipher Modes of Operation, Methods and Techniques, December 2001	NIST SP 800-38A
Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, May 2005	NIST SP 800-38B
Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality, July 2007	NIST SP 800-38C
Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC, November 2007	NIST SP 800-38D
Recommendation for Block Cipher Modes of Operation: the XTS-AES Mode for Confidentiality on Storage Devices, January 2010	NIST SP 800-38E
Recommendation for Key-Derivation Methods in Key-Establishment Schemes, Revision 3, April 2018	NIST SP 800-56A
Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher, Revision 2, November 2017	NIST SP 800-67
Recommendation for Random Number Generation Using Deterministic Random Bit Generators, June 2015	NIST SP 800-90A
Recommendation for Key Derivation Using Pseudorandom Functions (Revised), October 2009	NIST SP 800-108
Recommendation for Cryptographic Key Generation, Revision 2, June 2020	NIST SP 800-133
Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1, February 2003	RFC 3447
Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation, March 2010	RFC 5639
Elliptic Curves for Security, January 2016	RFC 7748
A proposal for: Functionality classes for random number generators, Version 2.0, September 2011	BSI-AIS20/AIS31

Title	Number
Cryptographic Mechanisms: Recommendations and Key Lengths, Version 2021-01, March 2021	BSI TR-02102-1
Resources	
Security IC Platform Protection Profile with Augmentation Packages, Version 1.0; [ICPP]	BSI-CC-PP-0084-2014
Java Card Protection Profile – Open Configuration, Version 3.0.5, [JCSPP]	BSI-CC-PP-0099-2017
User Guidances	
Secure Processor Unit (SPU) – Anti-replay Island (ARI) Overview for SM8450	80-11140-16, Revision AC
SM8450/SM8475 Secure Boot Enablement	80-PV345-14, Revision AE
Qualcomm® Secure Processing Unit Enablement for SM8450/SM8475 Devices – User Guide	80-PV345-150, Revision AE
SM8450/SM8475 Security Guidance for Secure Processing Unit Application Developers	80-PV345-152, Revision AD
SM8450/SM8475 Secure Processor Unit SDK – API Reference	80-PV579-11, Revision AC
Qualcomm® Snapdragon™ Secure Processing Unit (SPU) Application Development User Guide	80-NU430-7, Revision AB
SMT Assembly Guidelines	SM80-P0982-1, Revision E

B.2 Acronyms and terms

Acronym or term	Definition
AES	Advanced Encryption Standard
AIS	Application Notes and Interpretation of the Scheme
API	Application programming interface
ARI	Anti-replay island
BSI	Bundesamt für Sicherheit in der Informationstechnik (German: Federal Office for Security)
CBC	Cipher block chaining
CC	Common Criteria
CCM	Counter with CBC-MAC
CMAC	Cipher-based Message Authentication Code
CSR	Configuration and status register
CTR	Counter
DES	Data Encryption Standard
DDR	Double Data Rate
ECB	Electronic codebook
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
FIPS	Federal Information Processing Standards
FCI	Fuse Configuration Interface
GCC	Global Clock Controller

Acronym or term	Definition
GCM	Galois counter mode
HLOS	High Level Operating system
HMAC	Hash-based Message Authentication Code
HW	Hardware
IC	Integrated circuit
KDF	Key derivation function
MCP	Main control program
NIST	National Institute of Standards and Technology
NOC	Network on chip
NVM	Non-volatile memory
OAEP	Optimal Asymmetric Encryption Padding
OEM	Original equipment manufacturer
OS	Operating system
OSAT	Outsourced Semiconductor Assembly and Test
OTP	One-time programmable
PBL	Primary Boot Loader
PCB	Printed circuit board
PKCS	Public Key Cryptography Standard
PoP	Package-on-package
PP	Protection Profile
PSS	Probabilistic Signature Scheme
QTI	Qualcomm Technologies Inc.
RAM	Random access memory
RFC	Request for Comments
RNG	Random number generator
ROM	Read only memory
RPM	Resource and power manager
RSA	Rivest, Shamir, Adleman
RTL	Register transfer level
SFP	Security function policy
SFR	Security functional requirement
SHA	Secure Hash Algorithm
SIM	Subscriber Identity Module
SoC	System-on-chip
SP-AOTimer	Always-on timer
SP-AR	Anti-replay controller
SP-CE	Crypto engine
SP-CMC	Crypto management controller
SP-CMU	Crypto management unit
SP-CPU	Central processing unit
SP-DMA	Direct memory access
SP-ExtMM	External memory manager

Acronym or term	Definition
SP-KT	Key table
SP-LRM	Local resource manager
SP-MMU	Memory management unit
SP-PKA	Asymmetric crypto operation
SP-QFPROM	Qualcomm Fuse-Programmable Read-Only Memory
SP-SC	Security controller
SP-sMEM	Shared memory
SP-sCSR	Shared configuration and status registers
SPU	Secure Processor Unit
ST	Security Target
SW	Software
TEE	Trusted Execution Environment
TIC	Test interface controller
TME	Trusted Management Engine
TOE	Target of evaluation
TR	Technical report
TSF	TOE security function
TSS	TOE summary specification
xPU	External protection unit; (x is memory/register/address)