MSP V1.0 on Kirin 9000 Series Security Target

Issue 12

Date 2020-08-10

HiSilicon Technologies Co., Ltd.

Address: Huawei Industrial Base
Bantian, Longgang
Shenzhen 518129
People's Republic of China

Website: http://www.hisilicon.com/cn/

Email: support@hisilicon.com

Contents

Figures

Tables

1 ST Introduction

1.1 ST Reference

Title MSP V1.0 on Kirin 9000 Series Security Target

Version V12

Author Hisilicon

Date of publication August 10, 2020

Evaluation lab Brightsight

1.2 TOE Reference

TOE Name MSP

TOE Version V1.0

TOE Developer Hisilicon

1.3 TOE Overview

1.3.1 TOE Type

The TOE type is independent subsystem that is integrated in a system-on-chip (SoC) running closed security software.

1.3.2 TOE usage & Major Security Features

The TOE enables the use of mobile devices for system services that require security protection, as following:

Secure Storage SA Service

This service can provide a secure storage allocation for some fixed TA, and provide to read and write function. It’s taken by TAs to use secure storage features to store highly secure and confidential data.

Anti-Rollback SA Service

This service can provide anti-rollback counter storage function. The TOE can open the counter resource allocation for the TAs, and the counter read-write interface is used for the TAs to implement the anti-rollback feature.

Root of Trust SA Service

This service can provide root of trust function, include the certification of phone root of trust, key management and cryptographic service.

Weaver Authentication SA Service

This service can provide weaver authentication mechanism for mobile user authentication. If the weaver or biometric authenticates succeed, the user just can use the mobile devices.

File Encryption SA Service

This service can provide a security mechanism to store the file encryption different Class keys. Just only weaver or biometric SA identify successfully, the user can use these keys.

Biometric Authentication Supporting Service for TSA

This service can provide security API for the third party's biometric SA to support biometric authentication.

The TOE implements the following security features:

1. Cryptographic supports and random number generation
2. Physical protection against non-invasive, semi-invasive, and invasive physical attacks
3. Security update and boot
4. TOE's security storage
5. SA management
6. TOE's security runtime
7. Security functions for PSA services

1.3.3 Non TOE Hardware/Software/Firmware

Type	Name	Version
Hardware	NVM	The Version is presented in《MSP V1.0 on Kirin 9000 Series Operational User Guidance V10》
	DDR	
	Secure Flash	
	Kirin SOC chip	
Firmware	ATF	
Software	TEE	

Application Note: The TOE can be integrated into any of the Kirin 9000 Series of SoC, but the TOE functionality is the same regardless of the Kirin variant. Additionally, the Kirin variants in this series are identical in terms of the SoC hardware and differ only in the software running on the SoC.

Application Note: The TOE does not include Third-Party SAs for biometric authentication. These need to be implemented by the user.

1.4 TOE Description

1.4.1 TOE Logical Scope

The TOE is comprised of several security features. Each of the security features identified above consists of several security functionalities, as identified below:

1. Cryptographic supports and random number generation

   Generation of random numbers

   AES (128, 192 and 256 bit keys by ECB, CBC mode)

   RSA (2048 bit keys)

   Hash functions: SHA-256

   CMAC with AES using 128, 192 or 256 bit keys
2. Physical protection against non-invasive, semi-invasive, and invasive physical attacks

   Memory scrambling

   Side-channel analysis countermeasures

   Fault attacks and countermeasures

   Memory/registers integrity checking

   Package-on-package (PoP) form factor
3. Security update and boot

   Secure boot and secure loading of TOE software stored out of TOE, the TOE enforces the initialization of the TSF through a secure process including the verification of the authenticity, confidentiality and integrity.

   Secure update of the TOE software, the TOE enforces the update of the TSF by decryption and verification
4. TOE's security storage

   The TOE connects a Secure Flash by SCP03 for storing TOE’s user data and TOE’s TSF data’s mac.

   The TOE provides a security mechanism for storing TOE’s software, PSA code and data, and TSA data and code out of the TOE including confidentiality and integrity. At the same time, the TOE provides a policy about isolating the different SAs and SEE from each other.
5. SA management

   The TOE provides a series of system API with permission control. The SA just can access authorized API.

   The TOE loads the PSA by security mechanism for confidentiality and integrity.

   The TSA is loaded from out of the TOE for enforcing decryption and verification.

   The SA must access its own execution and storage space, which is separated from the spaces of any other SA.
6. TOE’s security runtime

   The TOE prevents the SAs from accessing the software own execution and storage space and resources.

The TOE shall ensure that runtime data, the SA code and the SA data and keys are protected against unauthorized modification and disclosure at runtime.

The TOE distinguishes between privilege level and user level, and different level commands just call owner level functions.

7. Security service
   - Secure Storage SA

     This SA provides access control for operating different TA's storage space data.
   - Anti-rollback SA

     This SA provides access control for operating different TA's Monotonic-Counter for TA's version anti-rollback.
   - Root of Trust SA

     This SA provides the operation of the device Root of Trust, key management and cryptographic function
   - Weaver Authentication SA

     This SA provides weaver data authentication error penalty function.
   - File Encryption SA

     This SA provides file encryption key storage function with password/biometric identification protection.
   - Biometric Authentication Supporting Service for TSA

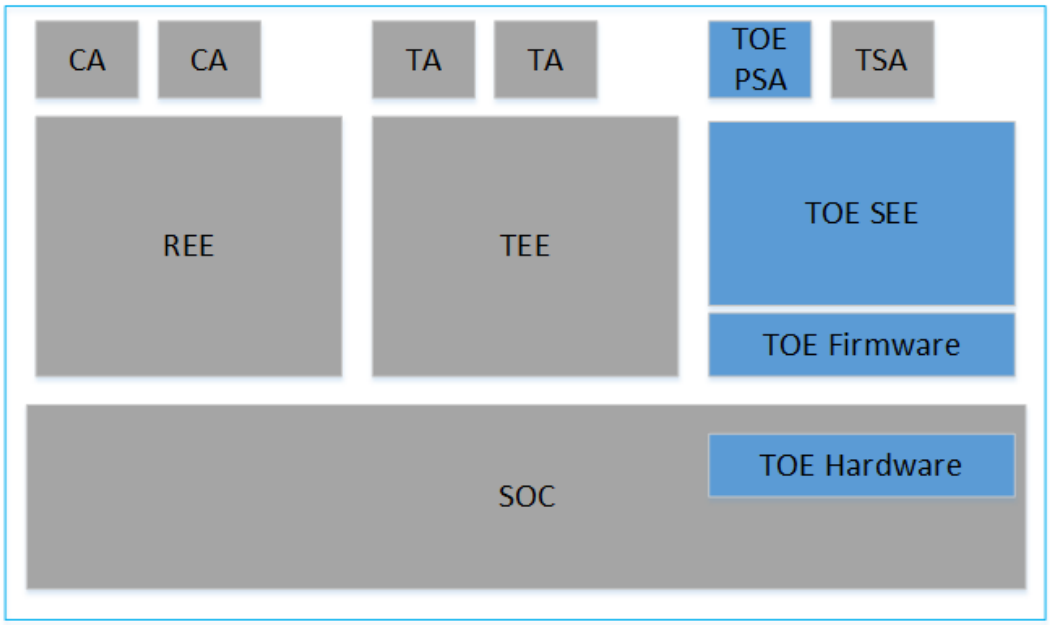
     This service provides TSA's data secure storage, for instance, biometric template authentication data and number of retries for failed biometric matching.

1.4.2 TOE Physical Scope

1.4.2.1 The architecture of SOC

The system security architecture diagram of the SOC is as follows Figure 1-1， where the CA of the related system security service calls TA, Then this TA calls the related SA to achieve high security. The TOE is composed of the hardware, firmware, software (SEE (include Framework and Core) and PSA), the following content will introduce the detailed information.

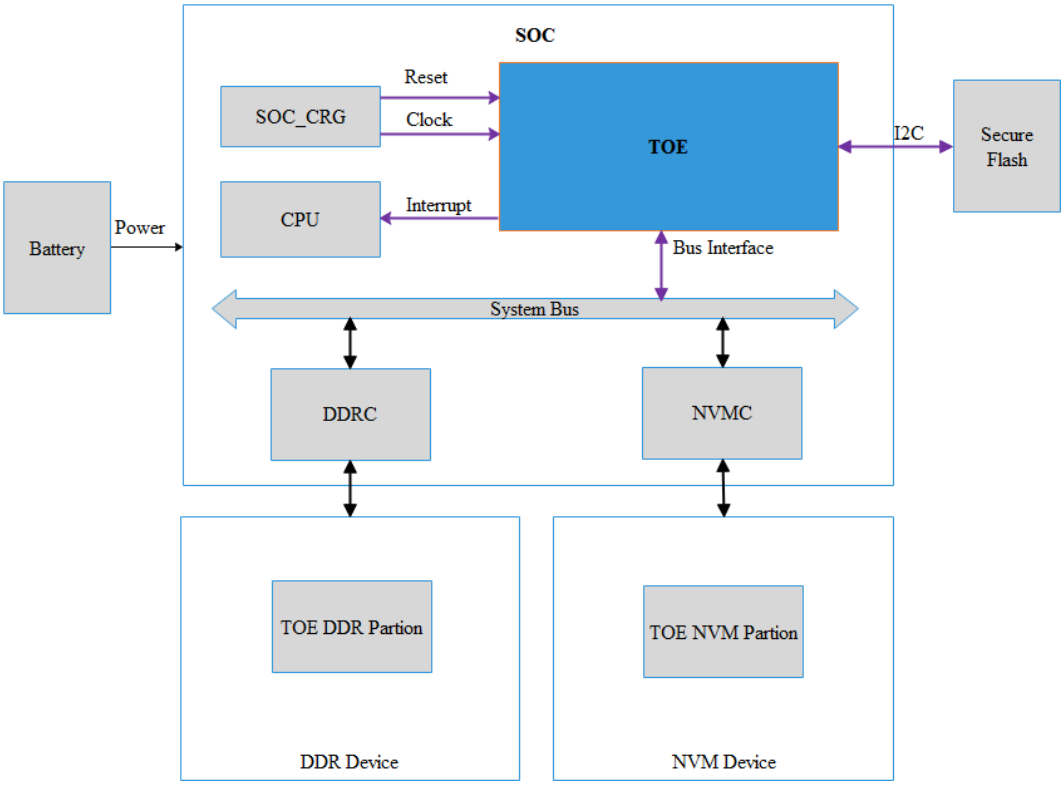
Figure 1-1 SoC's system security architecture



The TOE hardware interfaces to the SOC into which it is integrated are shown in Figure 1-2. The TOE hardware interfaces are:

- Clock and reset source, for TOE receive internal clock and internal reset regenerate.
- System access to allow TOE to read/write via Bus Interface in the TOE shared memory in DDR and NVM, such as flash device.
- Interface to access external secure component (such as secure flash) via the I2C bus.
- The interrupt interface between the TOE and external CPU is interrupt interface, use to deal with some abnormal response.

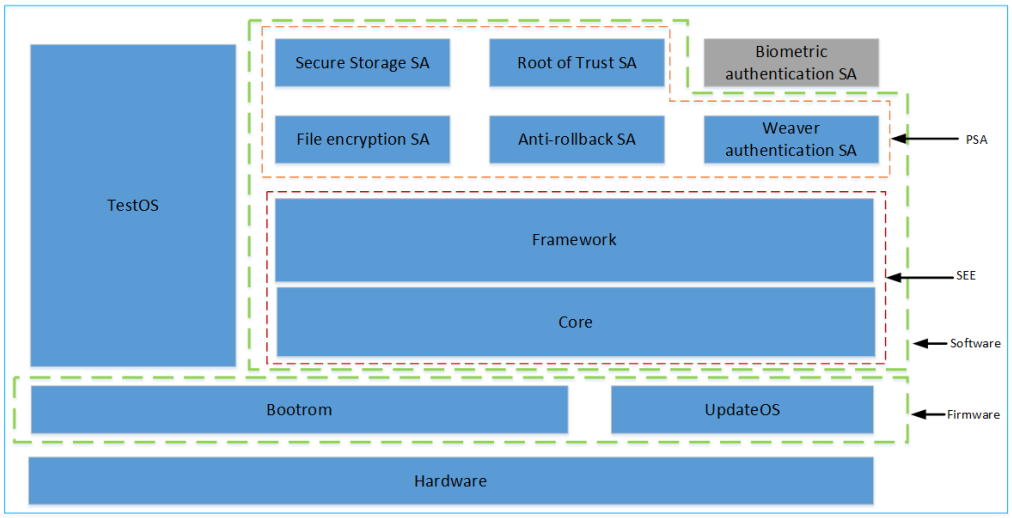
Figure 1-2 TOE components and their interfaces to the SoC



1.4.2.2 The architecture of TOE

The architecture of TOE is described as following Figure 1-3:

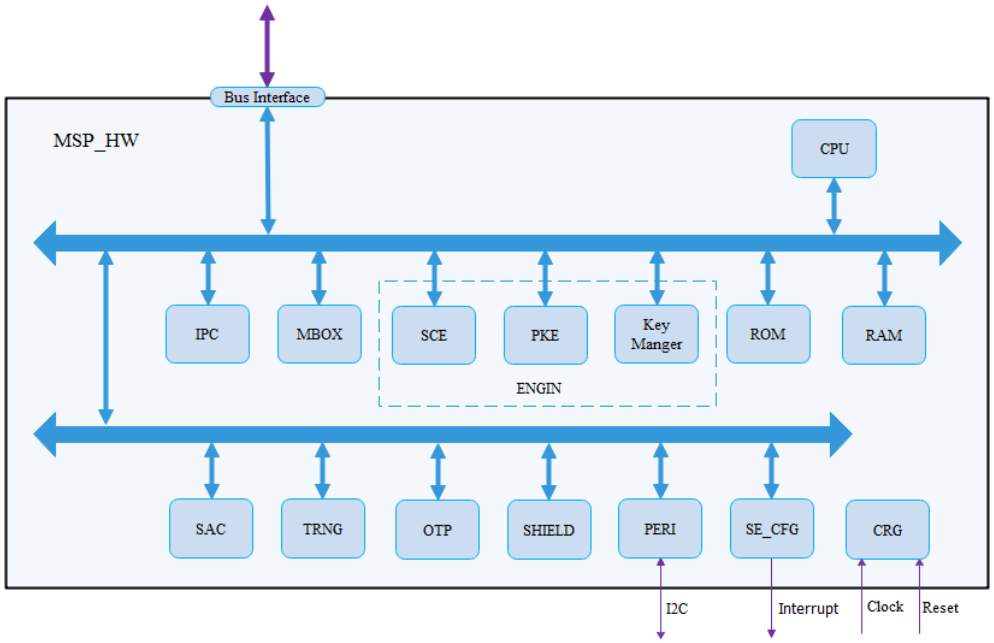
Figure 1-3 TOE Hardware, Firmware, Software architecture



TOE Hardware

The TOE contains TOE hardware described as following Figure 1-4:

Figure 1-4 TOE hardware components



TOE and other system via Bus interface, Bus interface will connect the inner of the TOE to SOC.

Bus is the main data path, it can transfer data out of the TOE and receive data from the outside of the TOE, TOE via the IPC and MBOX to interaction data with SOC other system, The ENGIN, include SCE, PKE, Key manager, use the bus for data transfer, CPU via bus to get ROM data when the TOE boot up, and via bus for reading or writing some data to the RAM.

Some peripheral components are accessed by the bus:

SAC is used for the user access authority.

TRNG generate the true RNG data for all the TOE which component need.

OTP is storing the key and some important information of the TOE, which should be not lost while power off.

SE_CFG mainly does some system configuration and sends some interrupt for the external CPU out of TOE.

SHIELD is an active shield for protecting against probe attack.

PERI subsys includes the secure timer, watch dog, I2C interface.

CRG is the clock and reset generate component for the TOE.

TOE Firmware

The TOE contains firmware, it includes Bootrom, which load TOE SEE for the secure boot process, and UpdateOS, which is used to load the updated software. Bootrom is stored in TOE ROM and UpdateOS is stored encrypted in NVM.

TOE Software

The TOE contains TestOS, it’s used by TOE Hardware manufacture for testing TOE and loading the initialization data or keys.

The TOE also contains software, it includes TOE SEE and PSA. The TOE's software' executable code is security stored in external NVM and DDR.

The SEE manages access to the services provided by the TOE, implements software countermeasures, and controls the applications. The SEE includes Framework level and Core level, the Core level can provide IO, I2C, security flash and other drivers, at the same time, can provide higher privilege. The framework level can provide SA manage, Access control and other functions, it also provides the system API for different SAs.

The PSA can provide different security service described 1.3.2 TOE usage & Major Security Features

1.4.2.3 TOE components

Table 1-1 TOE components

Type	Name	Version	Form of Delivery	Method of Delivery
Hardware	MSP_HW	V100	GDS that has been integrated in the SOC	Huawei internal delivery tool
Firmware	Bootrom	V100	GDS that has been integrated in the SOC	Huawei internal delivery tool
	UpdateOS	V100	Software image encrypted and signed	Huawei internal delivery tool
Test-software	TestOS	V100	Software image encrypted and signed	Huawei internal delivery tool
Software	SEE	V100R001	Software image encrypted and signed	Huawei internal delivery tool
	Secure Storage SA	V100R001		
	Anti-rollback SA	V100R001		
	Root of Trust SA	V100R001		
	Weaver authentication SA	V100R001		
	File encryption SA	V100R001		
Documents	MSP V1.0 on Kirin 9000 Series Preparative Procedures for	V10	PDF	GPG+email

Type	Name	Version	Form of Delivery	Method of Delivery
	User			
	MSP V1.0 on Kirin 9000 Series Operational User Guidance	V10	PDF	GPG+email
	MSP V1.0 IPC Command User Guidance	V01	PDF	GPG+email
	Secure Flash User Guidance	V01	PDF	GPG+email
	SA Development User Guidance	V02	PDF	GPG+email
	Manufacture User Guidance	V01	PDF	GPG+email

1.4.3 TOE Life Cycle

Figure 1-5 TOE Life Cycle

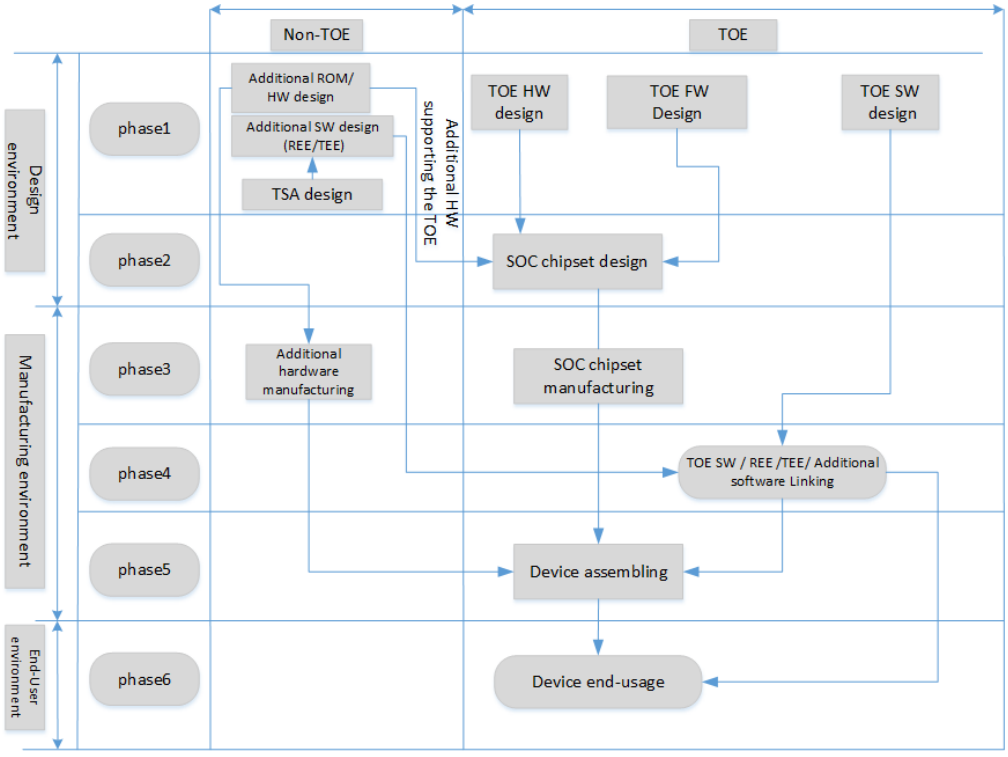


Table 1-2 TOE Life Cycle

Phases	Actors
1：Firmware/Software/Hardware/SA design	The TOE hardware designer is in charge of designing the TOE’s processor of SOC, where the TOE software runs. The TOE firmware designer is in charge of designing the TOE's security boot. The hardware and firmware will be linked with the whole SOC in phase 2. The TOE software designer is in charge of designing the TOE’s SEE and PSA to provide the security services to SOC. This parts will be linked in phase 4.
2: SOC chipset Linking	The SOC designer is in charge of Linking the all processors for Designing a GDS file and send it to chip manufacturing in phase 3.
3: SOC chipset manufacturing	The manufacture vendor produces the SOC chipset and enables, sets or seeds a series of security actions. The SOC chip will be sent to the Device manufacturing in phase 5.
4: Software manufacturing	The device manufacturer is responsible for the integration, validation and preparation of the software to load in the product that will include The TOE's software and additional software required to use the product (e.g. REE, TEE including the third party SA). The linked package will be sent to Device manufacturing in phase 5. If the Device has been in Device end-usage phase, the linked updated package will be sent by the OTA.
5: Device manufacturing	The device manufacturer is responsible for the device assembling and initialization and any other operation on the device (including loading or installation of TOE) before delivery to the end-user in phase 6.
6: Device end-usage	The end user gets a device ready for using.

Application Note:

The TOE operational phase starts in Phase 4.

2 Conformance Claims

2.1 CC Conformance claim

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.

2.2 PP Claim

This Security Target does not claim conformance with any protection profile.

2.3 Package Claim

The Security Target claims conformance to EAL5 augmented by ALC_DVS.2 and AVA_VAN.5.

2.4 Conformance Claim Rationale

3 Security Problem Definition

3.1 Definition of assets and TSF data

The assets to be protected are as follows:

1. User data
   - General data
     - Mobile phone Root of Trust
   - Data provided by Trusted Applications to Platform Secure Applications
     - Weaver value
     - count value
     - File Encryption Keys
     - Security Critical Data
   - Third-party Secure Applications
     - Runtime code
     - data (e.g. biometric template)
2. Security services
   - provided to third-party Secure Applications
     - Secure storage of biometric template authentication data
     - Secure storage of Number of retries for failed biometric matching
   - provided to Trusted Applications by Platform Secure Applications
     - Secure storage of the Security Critical Data
     - File Encryption Key derivation
     - Monotonic counter
     - Key, ID attestation and key management based on the mobile phone Root of Trust
     - cryptographic services
     - Weaver value verification
     - Authentication-dependent export of File Encryption Keys

The TSF data to be protected are as follows:

1. TOE’s hardware TSF data
   - The data of the TOE hardware

- TestOS code and data

2. TOE's software TSF data
   - TOE firmware code
   - TOE software code
   - TOE runtime data
   - TOE persistent data
   - TOE storage Root of Trust

3.2 Threats to Security

This section identifies the threats to assets that require protection by the TOE. The threats are defined in terms of assets concerned, attackers and the adverse action that materializes the threat.

Table 3-1 Threats

Threat	Description
T.Leak-Inherent	An attacker may illegal exploit information which is leaked from the TOE during usage of the TOE in order to disclose user data and TSF data.
T.Phys-Probing	An attacker may perform physical probing of the TOE in order • to disclose user data while stored in protected memory areas • to disclose/reconstruct the user data while processed or • to disclose other critical information about the operation of the TOE to enable attacks disclosing or manipulating the user data of the TOE.
T.Malfunction	An attacker may cause a malfunction of the TOE by applying environmental stress in order to • modify security services of the TOE or • modify functions of the TOE • deactivate or affect security mechanisms of the TOE to enable attacks disclosing or manipulating the user data of the TOE. This may be achieved by operating the TOE outside the normal operating conditions.
T.Phys-Manipulation	An attacker may physically modify the TOE in order to • modify user data of the TOE, • modify TSF data of the TOE, • modify or deactivate security services of the TOE, or • modify security mechanisms of the TOE to enable attacks disclosing or manipulating the user data of the TOE.
T.Leak-Forced	An attacker may exploit information which is leaked from the TOE during usage of the TOE in order to disclose confidential user data of the TOE as part of the assets even if the information leakage is not inherent but caused by the attacker.

Threat	Description
T.Abuse-Func	An attacker accesses TOE's functionalities outside of their expected availability range thus violating irreversible phases of the TOE life cycle or state machine. An attacker manages to instantiate an illegal TOE or to start-up the TOE in an insecure state or to enter an insecure state, allowing the attacker to obtain user and TSF data or compromise the TSF (bypass, deactivate or change security services).
T.RND	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.
T.Clone	An attacker manages to copy TOE related data of a first device on a second device and makes this device accept them as genuine data.
T.Memory_Dump	An attacker partially or totally recovers the content of the external NVM or DDR in plaintext, thus disclosing TOE's software code, potentially allowing the attacker to mount other attacks.
T.Impersonation	An attacker impersonates a SA to gain unauthorized access to the services and data of another Secure Application.
T.Rogue_Code_Execution	An attacker imports malicious code into the TOE to disclose or modify sensitive data.
T.Perturbation	An attacker modifies the behavior of the TOE's software in order to disclose or modify sensitive data or to force the TOE's software to execute unauthorized services.
T.RAM	An attacker partially or totally recovers RAM content, thus disclosing runtime data and potentially allowing the attacker to interfere with the TOE runtime data and software code.
T.SPY	An attacker discloses confidential data or keys by means of runtime attacks or unauthorized access to storage locations.
T.Security-Critical-Data	An entity or an unauthorized user holding the physical TOE discloses, modifies, or rolls back TA Security Critical Data. TA can operate other TA's storage Data address space.
T.Monotonic-Counter	Any entity decreases any of the TA-specific monotonic counters provided by the corresponding platform SA. TA can operate other TA's Monotonic-Counter address space.
T.Phone-Root-of-Trust	Any external entity discloses the mobile phone Root of Trust, or directly forges signatures in order to perform key or ID attestation without authorization.
T.weaver	An entity or an unauthorized user holding the physical TOE discloses, modifies, or rolls back any TA-provided weaver values, or discloses the corresponding PIN through a brute-force attack.
T.File-Encryption	An entity or an unauthorized user holding the physical TOE discloses the File Encryption Keys or triggers the unauthorized release of the File Encryption Keys to the TEE, by forging weaver or biometric authentication tokens.

3.3 Organisational Security Policies

Table 3-2 OSPs

OSPs	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.Integration_Configuration	Integration and configuration of the TOE by the phone manufacturer shall rely on guidelines defined by the TOE provider, which states all the security requirements for the phone manufacturer issued from the TOE evaluation.
P.Secrets	Generation, storage, distribution, destruction, injection of secret data in the TOE or any other operation performed outside the TOE shall enforce integrity and confidentiality of these data. This applies to secret data injected before end-usage phase (such as the root of trust of TOE storage) or during the end-usage phase (such as cryptographic private or symmetric keys, confidential data).
P.Third-Party-SA-API	The TOE provides the following services to third-party SAs: Secure storage of biometric templates authentication data and try counters.
P.Update	The TOE provides TOE software update services to device user.

3.4 Assumptions

The assumptions when using the TOE are the following:

Table 3-3 Assumptions

Assumption	Description
A.Protection_After_Delivery	It is assumed that the TOE is protected by the environment after delivery and before entering phase4 and phase5. It is assumed that the persons manipulating the TOE in the operational environment apply the TOE guidelines (e.g. user and administrator guidance, installation documentation, personalization guide). It is also assumed that the persons responsible for the application of the procedures contained in the guides, and the persons involved in delivery and protection of the product have the required skills and are aware of the security issues.
A.SA-Development	Third-party SA developers are assumed to comply with the third-party SA development guidelines set by the TOE provider.

Assumption	Description
A.Secure-Component	It is assumed that the TOE is connected to a Secure Flash component, which is another trusted IT security product that supports a secure channel to the TOE and protects the data stored inside from rollback.

4 Security Objectives

4.1 Security Objectives for the Operational Environment

The security objectives for the Operational Environment determine the responsibility of the environment in countering the threats, enforcing upholding the assumptions. Each objective must be traced back to aspects of identified threats to be countered by the environment.

Table 4-1 Security Objectives for the Operational Environment

Security Objective	Description
OE.Process-Sec-IC	Security procedures shall be used after TOE' hardware Delivery up to delivery to the phone manufacturer 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).
OE.Integration_Configuration	Integration and configuration of the TOE's software by the phone manufacturer shall rely on guidelines defined by the TOE's software provider, which states all the security requirements for the phone manufacturer issued from the TOE evaluation.
OE.Protection_After_Delivery	The TOE shall be protected by the environment after delivery and before entering the final usage phase. The persons manipulating the TOE in the operational environment shall apply the TOE guidance (e.g. user and administrator guidance, installation documentation, personalization guide).The persons responsible for the application of the procedures contained in the guides, and the persons involved in delivery and protection of the product have the required skills and are aware of the security issues.
OE.Secrets	Management of secret data (e.g. generation, storage, distribution, destruction, loading into the product of cryptographic private keys, symmetric keys, user authentication data) performed outside the TOE shall enforce integrity and confidentiality of these data.
OE.SA_Development	Third-party SA developers shall comply with the third-party SA development guidelines set by the TOE provider.
OE.Secure-Component	The phone manufacturer shall connect the TOE to a Secure Flash module, which is another trusted IT security product that supports a secure channel to the TOE and protects the data stored inside from

Security Objective	Description
	rollback.
OE.TEE-Logical	The TEE shall ensure that it only provides genuine TA identities to the TOE when requesting SA services and that it does not provide any logical interface to the TOE for any unauthorized entities.
OE.TEE-Interface	The environment shall include sufficient countermeasures to prevent any entity other than the TEE from gaining physical access to the interface between TEE and TOE.

4.2 Security Objectives for the TOE

The security objectives for the TOE must determine (to the desired extent) the responsibility of the TOE in countering the threats. Each objective must be traced back to aspects of identified threats to be countered by the TOE.

Table 4-2 Security Objectives for the TOE

Security Objective	Description
O.Identification	The TOE must provide means to store Initialization Data in its non-volatile memory. The Initialization Data (or parts of them) are used for TOE. The user shall be able to uniquely identify the TOE.
O.Leak-Inherent	Protection against Inherent Information Leakage The TOE must provide protection against disclosure of confidential data stored and/or processed • 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 The TOE must provide protection against disclosure/reconstruction of user data while stored in protected memory areas and processed 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 be

Security Objective	Description
	able to derive detailed design information or other information which 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 has not been proven or tested. This is to prevent malfunctions.
O.Phys-Manipulation	Protection against Physical Manipulation The TOE must provide protection against manipulation of the 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 TOE must be protected against disclosure of confidential data processed in the TOE (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 and/or • by a physical manipulation (refer to “Protection against Physical Manipulation (O.Phys-Manipulation)”).
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 in order to • disclose critical user data of the TOE, • manipulate critical user data of the TOE or, • 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.RND	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 since they might be used for instance to generate cryptographic keys.
O.Initialization	The TOE shall be started through a secure initialization process that ensures: the integrity and confidentiality of the TOE initialization code and data.
O.Secure_Load_ACode	The Loader of the Initial TOE shall check an evidence of authenticity and integrity of the loaded Additional Code.

Security Objective	Description
	The Loader enforces that only the allowed version of the Additional Code can be loaded on the Initial TOE. The Loader shall forbid the loading of an Additional Code not intended to be assembled with the Initial TOE. During the Load Phase of an Additional Code, the TOE shall remain secure.
O.Secure_AC_Activation	Activation of the Additional Code and update of the Identification Data shall be performed at the same time in an Atomic way. All the operations needed for the code to be able to operate as in the Final TOE shall be completed before activation. If the Atomic Activation is successful, then the resulting product is the Final TOE, otherwise (in case of interruption or incident which prevents the forming of the Final TOE such as tearing, integrity violation, error case…), the Initial TOE shall remain in its initial state or fail secure.
O.Operation	The TOE shall ensure the correct operation of its security functions. In particular, the TOE shall Protect itself against abnormal situations caused by programmer errors or violation of good practices by SAs. Control the access to its services by SAs: The TOE shall check the validity of any operation requested from SA. Enter a secure state upon failure detection, without exposure of any sensitive data.
O.Runtime_Integrity	The TOE shall ensure that the TOE firmware, the TOE runtime data, the SA code and the SA data and keys are protected against unauthorized modification at runtime when stored in volatile memory.
O.Runtime_Confidentiality	The TOE shall ensure that confidential TOE runtime data and SA data and keys are protected against unauthorized disclosure. The TOE shall not illegally export any sensitive data, random numbers or secret keys. The TOE shall grant access to sensitive data, random numbers or secret keys only to authorized SAs The TOE shall clean up sensitive resources as soon as it can determine that their values are no longer needed.
O.SA_Isolation	The TOE shall isolate the SAs from each other: Each SA shall access its own execution and storage space, which is separated from the spaces of any other SA.
O.TOE_Data_Protection	The TOE shall ensure the authenticity and confidentiality of TOE persistent data. It additionally shall support the protocols required by OE.Secure-Component to protect the consistency and integrity of the TOE persistent data, and to prevent the rollback of TOE persistent data.
O.SEE_Isolation	The SEE shall prevent the SAs from accessing the TOE' SEE own execution and storage space and resources.
O.Secure_Storage	The TOE shall provide Secure Storage services for SEE and persistent SA general-purpose data such that: • The confidentiality of the stored data is enforced

Security Objective	Description
	• The integrity of stored data is enforced • The atomicity of the operations that modify the storage is enforced. The TOE Secure Storage shall be bound to the hosting device, which means that the storage space must be accessible or modifiable only by authorized SAs running on the same TOE and device as when the data was created.
O.Third-Party-SA	The TOE shall provide the integrity, authenticity and confidentiality to third-party SA.
O.SA.Secure-Storage	The Secure Storage SA shall provide Secure Storage for different TAs general purpose data and key material such that: • The confidentiality of the stored data and keys is enforced • The authenticity of the stored data and keys is enforced • The integrity (including roll-back protection) of each TA stored data and keys is enforced (by supporting the protocols required by OE.Secure-Component to protect the consistency and integrity of the SA-provided Secure Storage) The atomicity of the operations that modify the storage is enforced. The SA provided Secure Storage shall be bound to the hosting device, which means that the storage space must be accessible or modifiable only by the Secure Storage SA running on the same TOE and device as when the data was created.
O.SA.Anti-Rollback	The Anti-Rollback SA shall provide a set of monotonic counters corresponding to TAs requesting this service. Each TA can request to increment or to read its own counter, and no entity can decrement this counter.
O.SA.Mobile-Root-of-Trust	The Root of Trust SA shall provide key management, signing and verification services using a TOE specific signing key corresponding to a certificate that is signed by the phone manufacturer.
O.SA.Weaver	The SA shall provide a PIN digest comparison service that returns a TA-configured value in case the comparison is successful, and that prevents the comparison to be performed for a specific amount of time (at a minimum) when a specific number of failed authentications have occurred in a row.
O.SA.File-Encryption	The File Encryption SA shall provide an authentication-dependent key export service to provide the corresponding file encryption keys to the TEE after verifying that the authentication has been performed.

4.3 Security Objectives Rationale

4.3.1 Tracing of security objectives to SPD

The following table provides a mapping of security objectives tracing each security objective for the TOE back to threats countered by that security objective enforced by that security objective, and each security objective for the operational environment back to threats countered by that security objective, and assumptions upheld by that security objective. This illustrates that the security objectives counter all threats, the security objectives for the operational environment uphold all assumptions.

Tracing of security objectives to SPD

Security Objectives	**SPD**
O.RND	T.RND
O.Leak-Inherent	T.Leak-Inherent
O.Phys-Probing	T.Phys-Probing
O.Malfunction	T.Malfunction
O.Phys-Manipulation	T.Phys-Manipulation
O.Leak-Forced	T.Leak-Forced
O.Abuse-Func	T.Abuse-Func
O.Identification	P.Process-TOE P.Update
O.Operation	T. Abuse-Func T.Impersonation T.Rogue_Code_Execution T.Perturbation
O.Runtime_Integrity	T. Abuse-Func T.Clone T.Rogue_Code_Execution T.Perturbation T.RAM T.RNG T.SPY
O.Runtime_Confidentiality	T. Abuse-Func T.Clone, T.Rogue_Code_Execution, T.Perturbation T.RAM T.RNG, T.SPY

Security Objectives	SPD
O.SA_Isolation	T.Perturbation T.SPY T.RAM
O.Initialization	T.Leak-Inherent T.Clone T.Rogue_Code_Execution T.Perturbation
O.Secure_Load_ACode	T.Leak-Inherent T.Rogue_Code_Execution T.Perturbation P.Update
O.Secure_AC_Activation	T.Leak-Inherent T.Rogue_Code_Execution T.Perturbation P.Update
O.TOE_Data_Protection	T.Abuse-Func T.Clone, T.Rogue_Code_Execution, T.Perturbation, O.Secure_Storage
O.SEE_Isolation	T.Abuse-Func T.Perturbation, T.RAM T.SPY
O.Third-Party-SA	T.Secure_Storage T.Impersonation T.Rogue_Code_Execution P.Update
O.Secure_Storage	T.Clone T.Memory_Dump T.Rogue_Code_Execution
O.SA.Secure-Storage	T.Security-Critical-Data
O.SA.Anti-Rollback	T.Monotonic-Counter
O.SA.Mobile-Root-of-Trust	T.Phone-Root-of-Trust
O.SA.Weaver	T.weaver
O.SA.File-Encryption	T.File-Encryption

Security Objectives	SPD
OE.Process-Sec-IC	A.Process-Sec-IC
OE.Integration_Configuration	P.Integration_Configuration T.Rogue_Code_Execution
OE.Protection_After_Delivery	A.Protection_After_Delivery T.Rogue_Code_Execution
OE.Secrets	P.Secrets
OE.SA_Development	P.Third-Party-SA-API
OE.Secure-Component	A.Secure-Component
OE.TEE-Logical	T.Security-Critical-Data T.Monotonic-Counter
OE.TEE-Interface	T.Security-Critical-Data T.Monotonic-Counter

4.3.2 Justification of tracing

The following table maps the threats of the security problem established to the security objectives of the TOE and the security objectives of the operational environment.

Table 4-3 Threat Rationale

Threat	Security Objectives
T.RND	O.RNG ensures that random numbers are unpredictable, have sufficient entropy and are not disclosed.
T.Leak-Inherent	The objective O.Leak-Inherent directly covers this threat. O.Initialization ensures that TOE initialization is prevented from Leak-Inherent. O.Secure_Load_ACode and O.Secure_AC_Activation ensures that TOE update is prevented from Leak-Inherent.
T.Phys-Probing	The objective O.Phys-Probing directly covers this threat.
T.Malfunction	The objective O.Malfunction directly covers this threat.
T.Phys-Manipulation	The objective O.Phys-Manipulation directly covers this threat.
T.Leak-Forced	The objective O.Leak-Forced directly covers this threat.
T.Abuse-Func	The objective O.Abuse-Func directly covers this threat.
T.Clone	The combination of the following objectives ensures protection against cloning: O.Initialization ensures that the TOE's software is bound to the SOC of

Threat	Security Objectives
	the device functionalities or data used to detect or prevent cloning O.Secure_Storage ensures that the storage is bound to the device and prevents the TOE from using data that is inconsistent or not authentic.
T.Memory_Dump	The objective O.Secure_Storage ensures the confidentiality of the data stored in external memory.
T.Impersonation	The combination of the following objectives ensures protection against application impersonation attacks: O.Operation ensures the verification of SA before any operation on their behalf.
T.Rogue_Code_Execution	The combination of the following objectives ensures protection against import of malicious code: O.Initialization ensures that the TOE security functionality is correctly initialized and the integrity of TOE firmware O.Secure_Load_ACode and O.Secure_AC_Activation ensure that the TOE security functionality is correctly updated. O.Operation ensures correct operation of the security functionality O.Secure_Storage ensures protection of the storage from which code might be imported. OE.Integration_Configuration covers the import of malicious code in a phase other than the end-user phase OE.Protection_After_Delivery covers the import of malicious code in a phase other than the end-user phase.
T.Perturbation	The combination of the following objectives ensures protection against perturbation attacks: O.Initialization ensures that the TOE security functionality is correctly initialized O.Secure_Load_ACode and O.Secure_AC_Activation ensure that the TOE security functionality is correctly updated. O.Operation ensures correct operation of the security functionality and a proper management of failures O.Runtime_Confidentiality covers runtime TOE data which might influence the behavior of the TOE O.Runtime_Integrity ensures protection against unauthorized modification of security functionality at runtime O.SA_Isolation ensures the separation of the TA O.TOE_Data_Protection covers persistent TOE data which might influence the behavior of the TOE O.SEE_Isolation enforces the separation between the TOE and the outside.
T.RAM	The combination of the following objectives ensures protection against RAM attacks: O.Initialization ensures that the TOE security functionality is correctly initialized O.Runtime_Confidentiality prevents exposure of confidential data at

Threat	Security Objectives
	runtime O.Runtime_Integrity protects against unauthorized modification of code and data at runtime O.SA_Isolation provides a memory barrier between SAs O.SEE_Isolation provides a memory barrier between the OS and the SAs.
T.SPY	The combination of the following objectives ensures protection against disclosure: O.Runtime_Confidentiality ensures protection of confidential data at runtime O.SA_Isolation ensures the separation between SAs O.SEE_Isolation ensures that neither TEE nor SAs can access SEE data O.Secure_Storage ensures that data stored in the NVM/DDR/Secure Flash are accessible by the SA owner only.
T.Security-Critical-Data	The objective O.SA.Secure-Storage directly covers this threat. OE.TA-Identity and OE.TEE-Interface ensure that only legitimate users can access the corresponding SA functionality.
T.Monotonic-Counter	The objective O.SA.Anti-Rollback directly covers this threat. OE.TA-Identity and OE.TEE-Interface ensure that only legitimate users can access the corresponding SA functionality.
T.Phone-Root-of-Trust	The objective O.SA.Mobile-Root-of-Trust directly covers this threat.
T.weaver	The objective O.SA.Weaver directly covers this threat.
T.File-Encryption	The objective O.SA.File-Encryption directly covers this threat.

The following table maps the assumptions of the problem established to the security objectives of the TOE and the security objectives of the operational environment.

Table 4-4 Assumption Rationale

Assumption	Security Objectives
A.Protection_After_Delivery	The objective OE.Protection_After_Delivery directly covers this assumption.
A.SA-Development	The objective OE. SA-Development directly covers this assumption.
A.Secure-Component	The objective OE. Secure-Component directly covers this assumption.

Table 4-5 OSPs Rationale

OSP	Security Objectives
P.Process-TOE	The objective OE.Process-Sec-IC directly covers this OSP.
P.Integration_Configuration	The objective OE.Integration_Configuration directly covers this OSP.
P.Secrets	The objective OE.Secrets directly covers this OSP.
P.Third-Party-SA-API	The objective OE. SA-Development directly covers this OSP.
P.Update	The objective O.Secure_Load_ACode and O.Secure_AC_Activation and O.Third-Party-SA cover this OSP.

5 Extended Components Definition

5.1 Definition of the Family FCS_RNG

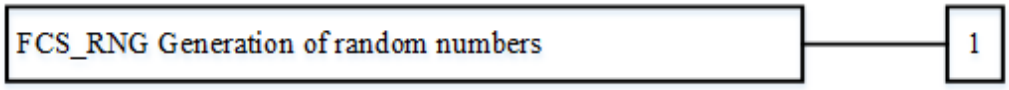
To define the IT security functional requirements of the TOE an additional family (FCS_RNG) of the Class FCS (cryptographic support) is defined here. This family describes the functional requirements for random number generation used for cryptographic purposes.

FCS_RNG Generation of random numbers

Family behaviour

This family defines quality requirements for the generation of random numbers which are intended to be use for cryptographic purposes.

Component levelling:



FCS_RNG.1 Generation of random numbers requires that random numbers meet a defined quality metric.

Management: FCS_RNG.1

There are no management activities foreseen.

Audit: FCS_RNG.1

There are no actions defined to be auditable.

FCS_RNG.1 Random number generation

Hierarchical to: No other components.

Dependencies: No dependencies.

FCS_RNG.1.1 The TSF shall provide a [selection: *physical, non-physical true, deterministic, hybrid physical, hybrid deterministic*] random number generator that implements: [assignment: *list of security capabilities*].

FCS_RNG.1.2 The TSF shall provide [selection: *bits, octets of bits, numbers [assignment: format of the numbers*]] that meet [assignment: *a defined quality metric*].

Application Note: A physical random number generator (RNG) produces the random number by a noise source based on physical random processes. A non-physical true RNG uses a noise

source based on non-physical random processes like human interaction (key strokes, mouse movement). A deterministic RNG uses a random seed to produce a pseudorandom output. A hybrid RNG combines the principles of physical and deterministic RNGs where a hybrid physical RNG produces at least the amount of entropy the RNG output may contain and the internal state of a hybrid deterministic RNG output contains fresh entropy but less than the output of RNG may contain.

5.2 Definition of the Family FMT_LIM

To define the IT security functional requirements of the TOE an additional family (FMT_LIM) of the Class FMT (Security Management) is defined here. This family describes the functional requirements for the Test Features of the TOE. The new functional requirements were defined in the class FMT because this class addresses the management of functions of the TSF. The examples of the technical mechanism used in the TOE appropriate to address the specific issues of preventing the abuse of functions by limiting the capabilities of the functions and by limiting their availability.

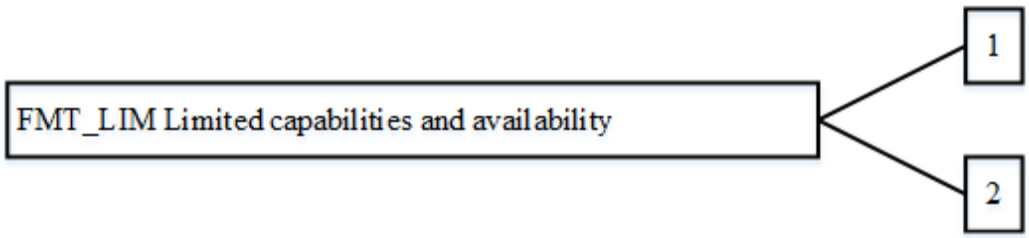
The family “Limited capabilities and availability (FMT_LIM)” is specified as follows.

FMT_LIM Limited capabilities and availability

Family behaviour

This family defines requirements that limit the capabilities and availability of functions in a combined manner. Note that FDP_ACF restricts the access to functions whereas the component Limited Capability of this family requires the functions themselves to be designed in a specific manner.

Component levelling:



FMT_LIM.1 Limited capabilities requires that the TSF is built to provide only the capabilities (perform action, gather information) necessary for its genuine purpose.

FMT_LIM.2 Limited availability requires that the TSF restrict the use of functions (refer to Limited capabilities (FMT_LIM.1)). This can be achieved, for instance, by removing or by disabling functions in a specific phase of the TOE’s life-cycle.

Management: FMT_LIM.1, FMT_LIM.2

There are no management activities foreseen.

Audit: FMT_LIM.1, FMT_LIM.2

There are no actions defined to be auditable.

The TOE Functional Requirement “Limited capabilities (FMT_LIM.1)” is specified as follows.

FMT_LIM.1 Limited capabilities

Hierarchical to: No other components.

Dependencies: FMT_LIM.2 Limited availability.

FMT_LIM.1.1 The TSF shall be designed and implemented in a manner that limits its capabilities so that in conjunction with “Limited availability (FMT_LIM.2)” the following policy is enforced [assignment: *Limited capability policy*].

The TOE Functional Requirement “Limited availability (FMT_LIM.2)” is specified as follows.

FMT_LIM.2 Limited availability

Hierarchical to: No other components.

Dependencies: FMT_LIM.1 Limited capabilities.

FMT_LIM.2.1 The TSF shall be designed in a manner that limits its availability so that in conjunction with “Limited capabilities (FMT_LIM.1)”the following policy is enforced [assignment: *Limited availability policy*].

5.3 Definition of the Family FAU_SAS

To define the security functional requirements of the TOE an additional family(FAU_SAS) of the Class FAU (Security Audit) is defined here. This family describes the functional requirements for the storage of audit data. It has a more general approach than FAU_GEN, because it does not necessarily require the data to be generated by the TOE itself and because it does not give specific details of the content of the audit records.

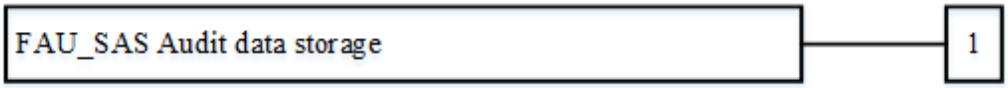
The family “Audit data storage (FAU_SAS)” is specified as follows.

FAU_SAS Audit data storage

Family behaviour

This family defines functional requirements for the storage of audit data.

Component levelling



FAU_SAS.1 Requires the TOE to provide the possibility to store audit data.

Management: FAU_SAS.1

There are no management activities foreseen.

Audit: FAU_SAS.1

There are no actions defined to be auditable.

FAU_SAS.1 Audit storage

Hierarchical to: No other components.

Dependencies: No dependencies.

FAU_SAS.1.1 The TSF shall provide [assignment: *list of subjects*] with the capability to store [assignment: *list of audit information*] in the [assignment: *type of persistent memory*].

5.4 Definition of the Family FDP_SDC

To define the security functional requirements of the TOE an additional family (FDP_SDC.1) of the Class FDP (User data protection) is defined here.

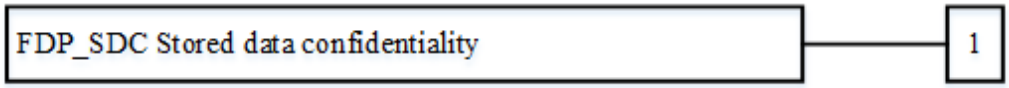
The family “Stored data confidentiality (FDP_SDC)” is specified as follows.

FDP_SDC Stored data confidentiality

Family behaviour

This family provides requirements that address protection of user data confidentiality 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 interfaces only and prevents compromise of their information bypassing these interfaces. It complements the family stored data integrity (FDP_SDI) which protects the user data from integrity errors while being stored in the memory.

Component levelling



FDP_SDC.1 Requires the TOE to protect the confidentiality of information of the user data in specified memory areas.

Management: FDP_SDC.1

There are no management activities foreseen.

Audit: FDP_SDC.1

There are no actions defined to be auditable.

FDP_SDC.1 Stored data confidentiality

Hierarchical to: No other components.

Dependencies: No dependencies.

FDP_SDC.1.1 The TSF shall ensure the confidentiality of the information of the user data while it is stored in the [assignment: *memory area*].

5.5 Definition of the Family FPT_TUD

To define the security functional requirements of the TOE an additional family (FPT_TUD) of the Class FPT (Protection of the TSF) is introduced here. This family describes the functional requirements for the update of the TSF by a dedicated function of the TOE that ensures the update in a correct and secure operational state.

The family “TSF Update Verification (FPT_TUD)” is specified as follows.

FPT_TUD TSF Update Verification

Family behaviour

This family defines TOE’s update function

Component levelling:

FPT_TUD.1 Requires the TOE to provide a TSF Update function that brings the TSF into a secure operational state at update.

Hierarchical to: No other components.

Management: FPT_TUD.1

No management activities are foreseen.

Audit: FPT_TUD.1

No actions are defined to be auditable.

FPT_TUD.1 TSF Update Verification

Hierarchical to: No other components.

Dependencies: [FCS_CKM.1 Cryptographic key generation or FCS_CKM.2 Cryptographic key distribution] FCS_COP.1 Cryptographic operation.

FPT_TUD1.1 The TSF shall decrypt and verify TOE's software updates by the manufacturer trusted keys prior to installing those updates.

5.6 Definition of the Family FPT_INI

To define the security functional requirements of the TOE an additional family (FPT_INI) of the Class FPT (Protection of the TSF) is introduced here. This family describes the functional requirements for the initialization of the TSF by a dedicated function of the TOE that ensures the initialization in a correct and secure operational state.

The family "TSF Initialisation (FPT_INI)" is specified as follows.

FPT_INI TSF initialisation

Family behaviour

This family defines TOE's initialization function

Component levelling:

FPT_INI.1 Requires the TOE to provide a TSF initialization function that brings the TSF into a secure operational state at power-on.

Hierarchical to: No other components.

Management: FPT_INI.1

No management activities are foreseen.

Audit: FPT_INI.1

No actions are defined to be auditable.

FPT_INI.1 TSF initialization

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_INI.1.1 The TOE initialization function shall verify

the authenticity, confidentiality and integrity of TOE software code and data;

the version of the TOE software to prevent downgrade to previous versions

[assignment: *list of implementation-dependent verifications*]

prior to establishing the TSF in a secure initial state.

FPT_INI.1.2 The TOE initialization function shall detect and respond to errors and failures during initialization such that the TOE either successfully completes initialization or is halted.

FPT_INI.1.3 The TOE initialization function shall not be able to arbitrarily interact with the TSF after TOE initialization completes.

6 Security Requirements

6.1 Security Functional Requirements

6.1.1 Conventions

The following conventions are used for the completion of operations:

- ~~Strikethrough~~ indicates text removed as a refinement
- underlined text indicates additional text provided as a refinement.
- **Bold text** indicates the completion of an assignment.
- ***Italicised and bold text*** indicates the completion of a selection.
- Iteration/Identifier indicates an element of the iteration, where Identifier distinguishes the different iterations.

6.1.2 Definition of security policies

To avoid redundancy in the definition of SFRs, in this chapter the security policies are defined that have to be fulfilled by the TOE.

6.1.2.1 Definition

Subjects stand for active entities inside the TOE:

S.API: the SEE Internal API, with security attributes "caller" (SA identifier)

S.SA_INSTANCE: any SA instance with security attribute "SA_identity" (SA identifier)

S.SA_INSTANCE_SESSION: any session within a given SA instance, with security attribute "TA_identity" (TA identifier)

S.RAM_UNIT: RAM addressable unit, with security attribute "rights:(SA identifier/TEE) ->(Read/Write/ReadWrite/NoAccess). For instance, an addressable unit may be allocated or have its access rights changed upon SA instance creation or when sharing memory references between a client (TA, SA) and a SA.

Application note: A RAM_UNIT typically stands for a byte in the C language; and there is no RAM access restriction applicable to the SEE itself.

S.COMM_AGENT: proxy between TAs in the TEE and the SEE and its SAs.

S.TA_INSTANCE: any TA instance with security attribute "TA_identity" (TA identifier) and "TEE_identity"(TEE identifier).

Objects stand for passive entities inside the TOE:

OB.SA_STORAGE: Secure Storage space of a SA, with security attributes "owner" (SA identifier), "inExtMem" (True/False), "inSecMem" (True/False) and "SEE_identity" (SEE identifier).

OB.SEE_STORAGE: Secure Storage space of a SEE, with security attributes "owner" (SEE identifier), "inExtMem" (True/False), "inSecMem" (True/False) and "SEE_identity" (SEE identifier).

OB.SRT: the SEE Storage Root of Trust, with security attribute SEE_identity" (SEE identifier).

OB.API: the SEE Internal API, with security attributes "owner" (SA identifier or SEE identifier).

OB.SSSA_STORAGE: SecureFlash storage space of Secure Storage SA with security attribute "TA_identity"(TA identifier) and "TEE_identity"(TEE identifier).

OB.ATRSA_COUNTER: monotonic counter of Anti-rollback SA with security attribute "TA_identity" and "TEE_identity".

Operation:

Sescure Storage

OP.LOAD: any operation used to get back persistent objects to be used by the SA or SEE.

OP.STORE: any operation used to store persistent objects. It stands for object creation, object deletion, object renaming, object truncation and write to an object.

SEE API

OP.CALL_API: Any executing API operation.

Secure Storage SA

OP.SSSA_CREATE: any operation used to create persistent object (SecureFlash storage space) to be used by the TA.

OP.SSSA_DELETE: any operation used to delete persistent object (SecureFlash storage space) to be used by the TA.

OP.SSSA_LOAD: any operation used to get back persistent objects (SecureFlash storage space) to be used by the TA.

OP.SSSA_STORE: any operation used to store persistent objects (SecureFlash storage space). It stands for object creation, object deletion, object renaming, object truncation and write to an object.

Anti-rollback SA

OP.CREATE_COUNTER: any operation used to create a monotonic counter object to be used by the TA.

OP.STEP_COUNTER: any operation used to increase a monotonic counter object to be used by the TA.

OP.EXTRACT_COUNTER: any operation used to extract a monotonic counter object to be used by the TA.

Information stands for data exchanged between subjects:

I.RUNTIME_DATA (user data or TSF Data depending on the owner): data belonging to the SA or to the SEE itself. Stands for parameter values, return values, content of memory regions in plaintext.

Application note: Data that is encrypted and authenticated is not considered I.RUNTIME_DATA.

TSF data consists of runtime TSF data and TSF persistent data (also called SEE persistent data) necessary to provide the security services. It includes all the security attributes of users, subjects, objects and information.

6.1.2.2 Secure Storage Access Control SFP

Purpose: To control the access to SEE and SA storage where persistent SEE and SA code and data are stored, which is granted on behalf of the SEE or owner SA only. This policy also enforces the binding of SA secure storage to the SEE storage root of trust OB.SRT

Subjects:

S.API.

Objects:

OB.SA_STORAGE, OB.SEE_STORAGE, OB.SRT.

Security attributes:

S.API.caller,

OB.SA_STORAGE.owner, OB.SA_STORAGE.inExtMem, OB.SA_STORAGE.inSecMem, OB.SA_STORAGE.SEE_identity, OB.SEE_STORAGE.owner, OB.SEE_STORAGE.inExtMem, OB.SEE_STORAGE.inSecMem, OB.SEE_STORAGE.SEE_identity and OB.SRT.SEE_identity,

Operations: OP.LOAD, OP.STORE.

6.1.2.3 Runtime Data Information Flow Control SFP

Purpose: To control the flow of runtime data from and to executable entities and memory. This policy contributes to ensure the integrity and confidentiality of runtime data

Subjects: S.SA_INSTANCE, S.SA_INSTANCE_SESSION, S.API, S.COMM_AGENT, S.RAM_UNIT.

Information: I.RUNTIME_DATA.

Security attributes: S.SA_INSTANCE. SA_identity, S.SA_INSTANCE_SESSION. TA_identity, S.RAM_UNIT.rights and S.API.caller.

6.1.2.4 TOE’s SEE API Access Control SFP

Purpose: To control the access to SEE's system API for different SA's using

Subjects: S.SA_INSTANCE.

Objects: OB.API.

Security attributes: OB.API.owner and S.SA_INSTANCE. SA_identity.

Operations: OP.CALL_API.

6.1.2.5 TA Access Control SFP

Purpose: To control the access from TAs to SAs by whitelist function.

Subjects: S.TA_INSTANCE.

Objects: OB.SSSA_STORAGE, OB.ATRSA_COUNTER.

Security attributes:

S.TA_INSTANCE.TA_identity, S.TA_INSTANCE.TEE_identity,

OB.SSSA_STORAGE.TA_identity, OB.SSSA_STORAGE.TEE_identity,

OB.ATRSA_COUNTER.TA_identity, OB.ATRSA_COUNTER.TEE_identity.

Operations:

For security storage SA:

OP.SSSA_CREATE, OP. OP.SSSA_DELETE, OP. SSSA_STORE, OP. OP.SSSA_LOAD.

For Anti-rollback SA:

OP.CREATE_COUNTER, OP.STEP_COUNTER, OP.EXTRACT_COUNTER.

6.1.3 Hardware

FDP_ITT.1 Basic internal transfer protection

FDP_ITT.1.1 The TSF shall enforce the **[**assignment: **Runtime Data Information Flow Control SFP]** to prevent the **[**selection: ***disclosure, modification*****]** of user data when it is transmitted between 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 **[**selection: ***disclosure, modification*****]** when it is transmitted between separate parts of the TOE.

FMT_LIM.1 Limited capabilities

FMT_LIM.1.1 The TSF shall be designed and implemented in a manner that limits its capabilities so that in conjunction with "Limited availability (FMT_LIM.2)" the following policy is enforced **[**assignment: **Deploying Test Features after TOE Delivery does not allow user data of the 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 might enable other attacks].**

FMT_LIM.2 Limited availability

FMT_LIM.2.1 The TSF shall be designed in a manner that limits its availability so that in conjunction with "Limited capabilities (FMT_LIM.1)"the following policy is enforced **[**assignment: **Deploying Test Features after TOE Delivery does not allow user data of the 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 might enable other attacks].**

FAU_SAS.1/Hardware **Audit storage**

FAU_SAS.1.1/Hardware The TSF shall provide **[**assignment: **the TestOS before TOE delivery]** with the capability to store **[**assignment: **the Initialization Data or keys]** in the **[**assignment: **OTP]**.

FDP_SDC.1 Stored data confidentiality

FDP_SDC.1.1 The TSF shall ensure the confidentiality of the information of the user data while it is stored in the **[**assignment: **the TOE hardware boundary].**

FDP_SDI.2/Hardware **Stored data integrity monitoring and action**

FDP_SDI.2.1/Hardware The TSF shall monitor user data stored in containers controlled by the TSF for **[**assignment: **CRC parity check/Error Correcting Code and errors after partial power collapse using a checksum function]** on all objects, based on the following attributes: **[**assignment: **data stored in the TOE hardware boundary]**

FDP_SDI.2.2/Hardware Upon detection of a data integrity error, the TSF shall **[**assignment: **perform a reset]**.

FPT_PHP.3 Resistance to physical attack

FPT_PHP.3.1 The TSF shall resist **[**assignment: **physical manipulation and physical probing]** to the **[**assignment: **TSF]** by responding automatically such that the SFRs are always enforced.

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: **[**assignment: **exposure to operating conditions that are not detected according to the requirement Failure with reservation of secure state FPT_FLS.1].**

FPT_FLS.1/Hardware **Failure with preservation of secure state**

FPT_FLS.1.1/Hardware The TSF shall preserve a secure state when the following types of failures occur: **[**assignment: **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].**

6.1.4 Firmware and Software

6.1.4.1 Cryptography

FCS_RNG.1 Random number generation

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

(PTG.3.1) A total failure test detects a total failure of entropy source immediately when the RNG has started. When a total failure is 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 [selection: ***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 has started, and (ii) while the RNG is being operated. The TSF must not output any random numbers before the power-up online test and 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 quality of the raw random number sequence. It is triggered [selection: ***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 **[**selection: ***numbers*** [assignment: **in 32-bit blocks**]] that meet

[assignment:

(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 [assignment: **None**]**.**

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

FCS_COP.1/AES **Cryptographic operation**

FCS_COP.1.1/AES The TSF shall perform **[**assignment: **encryption and decryption]** in accordance with a specified cryptographic algorithm **[**assignment: **AES in ECB mode, CBC mode]** and cryptographic key sizes **[**assignment: **128 bits, 192 bits, 256bits]** that meet the following: **[assignment: Specification for the ADVANCED ENCRYPTION STANDARD (AES) (FIPS PUB 197), Recommendation for Block Cipher Modes of Operation, Methods and Techniques (NIST SP 800-38A)]**.

FCS_COP.1/RSA **Cryptographic operation**

FCS_COP.1.1/RSA The TSF shall perform **[**assignment: **signature-verification or en/decryption]** in accordance with a specified cryptographic algorithm **[**assignment: **RSA-signature verification with SHA-256 and RSA-PSS/ RSA-OAEP, RSA-PKCS1-v1_5 padding]** and cryptographic key sizes **[**assignment: **2048 bits]** that meet the following: **[assignment: PKCS#1 RSA Cryptography Specifications v2.2 RSASSA-PSS/ RSAES-OAEP, RSASSA-PKCS1-v1_5]**.

FCS_COP.1/CMAC **Cryptographic operation**

FCS_COP.1.1/CMAC The TSF shall perform [assignment: **message authentication code generation**] in accordance with a specified cryptographic algorithm [assignment: **CMAC using AES**] and cryptographic key sizes [assignment: **128 bits, 192 bits and 256 bits**] that meet the following: [assignment: **Specification for the ADVANCED ENCRYPTION STANDARD (AES) (FIPS PUB 197), and Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication (NIST SP 800-38B)**].

FCS_COP.1/Hash **Cryptographic operation**

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

FCS_CKM.1/SYM **Cryptographic key generation**

FCS_CKM.1.1/SYM The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **[**assignment: **CMAC AES-256]** and specified cryptographic key sizes **[**assignment: **128 bits, 192bits, 256 bits**] that meet the following: **[**assignment: **NIST SP 800-38B].**

FCS_CKM.1/RSA **cryptographic key generation**

FCS_CKM.1.1/RSA The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm [assignment: **RSA key generation**] and specified cryptographic key sizes [assignment: **2048 bits**] that meet the following: [assignment: **PKCS#1 RSA Cryptography Specifications v2.2**].

FCS_CKM.4 Cryptographic key destruction

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method [assignment: **overwriting with zeroes or random number]** that meets the following: [assignment: **None**].

6.1.4.2 Security Update

FPT_TUD.1 TSF Update Verification

FPT_TUD1.1 The TSF shall decrypt and verify TOE's software updates by the manufacturer trusted keys prior to installing those updates.

Application Note: TSA is loaded from another component firstly, it also has the same security requirement as software update.

FAU_SAS.1/ACode **Audit storage**

FAU_SAS.1.1/ACode The TSF shall provide [assignment: **the TOE developer**] with the capability to store [assignment: **TOE software version**] in the [assignment: **TOE software image**].

6.1.4.3 Security Boot

FPT_INI.1 TSF initialization Verification

FPT_INI.1.1 The TOE initialization function shall verify

the authenticity, confidentiality and integrity of TOE software code and data;

the version of the TOE software to prevent downgrade to previous versions

[assignment: **None]** prior to establishing the TSF in a secure initial state.

FPT_INI.1.2 The TOE initialization function shall detect and respond to errors and failures during initialization such that the TOE either successfully completes initialization or is halted.

FPT_INI.1.3 The TOE initialization function shall not be able to arbitrarily interact with the TSF after TOE initialization completes.

6.1.4.4 Security Storage

FDP_SDI.2/Software **Stored data integrity monitoring and action**

FDP_SDI.2.1/Software The TSF shall monitor user data stored in containers controlled by the TSF for **[**assignment: **any integrity errors]** on all objects, based on the following attributes: **[**assignment: **AES128-CMAC value for data store]** while data is stored in the NVM and DDR**.**

FDP_SDI.2.2/Software Upon detection of a data integrity error, the TSF shall **[**assignment: **prevent the data from being used by the TOE]**.

FTP_ITC.1 Inter-TSF trusted channel

FTP_ITC.1.1 The TSF shall provide a communication channel between itself and another trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure.

FTP_ITC.1.2 The TSF shall permit **[**selection: ***the TSF*****]** to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for **[**assignment:

Secure Storage of TA and TEE data via the Secure Storage SA,

Secure Storage of Counter via the Anti-rollback SA,

Secure Storage of ROT and Keys via the ROT SA,

Secure Storage of Weaver's secret data via the Weaver Verification SA,

Secure Storage of Keys via the File Encryption SA,

Storage of TOE's SEE and SA's Mac.

].

FDP_ACC.1/SS **Subset access control**

FDP_ACC.1.1/SS The TSF shall enforce the **[**assignment: **Secure Storage Access Control SFP]** on

[assignment:

- **Subjects: S.API**
- **Objects: OB.SA_STORAGE, OB.SEE_STORAGE, OB.SRT**
- **Operations: OP.LOAD, OP.STORE**

].

FDP_ACF.1/SS **Security attribute based access control**

FDP_ACF.1.1/SS The TSF shall enforce the **[**assignment: **Secure Storage Access Control SFP]** to objects based on the following:

[assignment:

- **Subjects (attributes): S.API (S.API.caller)**
- **Objects (attributes): OB.SA_STORAGE (OB.SA_STORAGE.owner, OB.SA_STORAGE.inExtMem, OB.SA_STORAGE.inSecMem, OB.SA_STORAGE.SEE_identity), OB.SEE_STORAGE(OB.SEE_STORAGE.owner, OB.SEE_STORAGE.inExtMem, OB.SEE_STORAGE.inSecMem, OB.SEE_STORAGE.SEE_identity) and OB.SRT(OB.SRT.SEE_identity)**

].

FDP_ACF.1.2/SS The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

[assignment:

- **OP.LOAD of an object from OB.SA_STORAGE or OB.SEE_STORAGE is allowed if the following conditions hold:**

  The operation is performed by S.API

  The load request comes from an instance of the owner of the secure storage space (S.API.caller = OB.SA_STORAGE.owner or OB.SEE_STORAGE.owner)

  OB.SA_STORAGE or OB.SEE_STORAGE is bound to the TOE storage root of trust OB.SRT (OB.SA_STORAGE.SEE_identity or OB.SEE_STORAGE.SEE_identity = OB.SRT.SEE_identity)

  If OB.SA_STORAGE or OB.SEE_STORAGE is located in external memory accessible to the SEE (OB.SA_STORAGE.inExtMem = True or OB.SEE_STORAGE.inExtMem = True) then the object is authenticated and decrypted before load

  If OB.SA_STORAGE or OB.SEE_STORAGE is located in Secure memory accessible to the SEE (OB.SA_STORAGE.inSecMem = True or OB.SEE_STORAGE.inSecMem = True) then the object is decrypted before load

- **OP.STORE of an object to OB.SA_STORAGE or OB.SEE_STORAGE is allowed if the following conditions hold:**

  The operation is performed by S.API

  The store request comes from an instance of the owner of the secure storage space (S.API.caller = OB.SA_STORAGE.owner or OB.SEE_STORAGE.owner)

  OB.SA_STORAGE or OB.SEE_STORAGE is bound to the SEE storage root of trust OB.SRT (OB.SA_STORAGE.SEE_identity or OB.SEE_STORAGE.SEE_identity = OB.SRT.SEE_identity)

  If OB.SA_STORAGE or OB.SEE_STORAGE is located in external memory accessible to the SEE (OB.SA_STORAGE.inExtMem = True or OB.SEE_STORAGE.inExtMem = True) then the object is signed and encrypted before storage.

  If OB.SA_STORAGE or OB.SEE_STORAGE is located in Secure memory accessible to the SEE (OB.SA_STORAGE.inSecMem = True or OB.SEE_STORAGE.inSecMem = True) then the object is encrypted before storage

].

FDP_ACF.1.3/SS The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: [assignment: **None**].

FDP_ACF.1.4/SS The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **[**assignment:

- **Any access to a secure storage attempted from S.API without valid caller (S.API.caller = undefined)**
- **Any access to a secure storage that was bound to a different SEE (OB.SA_STORAGE.SEE_identity or OB.SEE_STORAGE.SEE_identity different from OB.SRT.SEE_identity)**
- **Any access to a secure storage from a subject different from S.API**

].

FMT_MSA.3/SS **Static attribute initialization**

FMT_MSA.3.1/SS The TSF shall enforce the **[**assignment: **Secure Storage Access Control SFP]** to provide **[**selection, choose one of: ***restrictive*****]** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/SS The TSF shall allow the **[**assignment: **None]** to specify alternative initial values to override the default values when an object or information is created.

6.1.4.5 Security Runtime

FDP_IFC.2/Runtime **Complete information flow control**

FDP_IFC.2.1/Runtime The TSF shall enforce the **[**assignment: **Runtime Data Information Flow Control SFP]** on

[assignment:

Subjects: S.SA_INSTANCE, S.SA_INSTANCE_SESSION, S.API, S.COMM_AGENT, S.RAM_UNIT

Information: I.RUNTIME_DATA

]

and all operations that cause that information to flow to and from subjects covered by the SFP.

FDP_IFC.2.2/Runtime The TSF shall ensure that all operations that cause any information in the TOE to flow to and from any subject in the TOE are covered by an information flow control SFP**.**

FDP_IFF.1/Runtime **Simple security attributes**

FDP_IFF.1.1/Runtime The TSF shall enforce the **[**assignment: **Runtime Data Information Flow Control SFP]** based on the following types of subject and information security attributes:

[assignment:

Subjects (attributes):

S.SA_INSTANCE (S.SA_INSTANCE. SA_identity),

S.SA_INSTANCE_SESSION (S.SA_INSTANCE_SESSION. TA_identity),

S.RAM_UNIT (S.RAM_UNIT.rights),

S.COMM_AGENT（None） and

S.API (S.API.caller)

]

FDP_IFF.1.2/Runtime The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold:

[assignment:

- **Rules for information flow between S.SA_INSTANCE and S.RAM_UNIT:**

  Flow of I.RUNTIME_DATA from S.SA_INSTANCE to S.RAM_UNIT is allowed only if S.RAM_UNIT.rights(S.SA_INSTANCE) is Write or ReadWrite

  Flow of I.RUNTIME_DATA from S.RAM_UNIT to S.SA_INSTANCE is allowed only if S.RAM_UNIT.rights(S.SA_INSTANCE) is Read or ReadWrite

- **Rules for information flow from and to S.COMM_AGENT:**

  Flow of I.RUNTIME_DATA from S.COMM_AGENT to S.RAM_UNIT is allowed only if S.RAM_UNIT.rights(TEE) is Write or ReadWrite

  Flow of I.RUNTIME_DATA from S.RAM_UNIT to S.COMM_AGENT is allowed only if S.RAM_UNIT.rights(TEE) is Read or ReadWrite

- **Rules for information flow from and to S.API:**

  Flow of I.RUNTIME_DATA from S.API to S.RAM_UNIT is allowed only if S.RAM_UNIT.rights(S.API.caller) is Write or ReadWrite

  Flow of I.RUNTIME_DATA from S.RAM_UNIT to S.API is allowed only if S.RAM_UNIT.rights(S.API.caller) is Read or ReadWrite

].

FDP_IFF.1.3/Runtime The TSF shall enforce the **[assignment: None].**

FDP_IFF.1.4/Runtime The TSF shall explicitly authorise an information flow based on the following rules: **[assignment:**

- **Rules for information flow from and to S.SA_INSTANCE_SESSION:**

  Flow of I.RUNTIME_DATA that are parameter or return values is allowed between S.SA_INSTANCE_SESSION and S.COMM_AGENT

  Flow of I.RUNTIME_DATA that are parameter or return values is allowed between S.SA_INSTANCE_SESSION and S.API.

].

FDP_IFF.1.5/Runtime The TSF shall explicitly deny an information flow based on the following rules: **[assignment: Any information flow involving a TOE subject unless one of the conditions stated in FDP_IFF.1.1/1.2/1.3/1.4 holds.].**

FMT_MSA.3/Runtime **Static attribute initialization**

FMT_MSA.3.1/ Runtime The TSF shall enforce the **[**assignment: **Runtime Data Information Flow Control SFP]** to provide **[**selection, choose one of: ***restrictive*****]** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/ Runtime The TSF shall allow the **[**assignment: **None]** to specify alternative initial values to override the default values when an object or information is created.

FPT_FLS.1/Runtime **Failure with preservation of secure state**

FPT_FLS.1.1/Runtime The TSF shall preserve a secure state when the following types of failures occur: **[**assignment:

- **Device binding failure**
- **Cryptographic operation failure**
- **Invalid TA requests, in particular bad-formed requests**
- **Panic states**
- **SA code or SA data authenticity or consistency failure**
- **TOE data (in particular SA properties, TOE keys and all security attributes) authenticity or consistency failure**
- **TOE software integrity failure**
- **TOE initialization failure**
- **Unexpected commands in the current TOE state**

].

Application Note:

Device binding failure occurs when (part of) the stored data has not been bound by the same TOE

FDP_ACC.1/API **Subset access control**

FDP_ACC.1.1/API The TSF shall enforce the **[**assignment: **TOE's SEE API Access Control SFP]** on **[**assignment:

- **Subjects: S.SA_INSTANCE**
- **Objects: OB.API**
- **Operations: OP.CALL_API**

].

FDP_ACF.1/API **Security attribute based access control**

FDP_ACF.1.1/API The TSF shall enforce the **[**assignment: **TOE's SEE API Access Control SFP]** to objects based on the following: **[**assignment:

- **Subjects (attributes): S.SA_INSTANCE (S.SA_INSTANCE.SA_identity)**
- **Objects (attributes): OB.API (OB.API.owner)**

].

FDP_ACF.1.2/API The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: **[**assignment:

OP.CALL_API of an object from OB.API is allowed if the following conditions hold:

The operation is performed by S.SA_INSTANCE

The SA instance that requested the operation owns the API

(S.SA_INSTANCE. SA_identity = OB.API.owner)

].

FDP_ACF.1.3/API The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: **[**assignment: **None]**.

FDP_ACF.1.4/API The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **[**assignment:

- **Any call to a SEE API from a subject different from S.SA_INSTANCE**
- **Any call to a SEE API attempted from S.SA_INSTANCE without valid SA_identity (S.SA_INSTANCE.SA_identity = undefined)**

].

FMT_MSA.3/API **Static attribute initialization**

FMT_MSA.3.1/API The TSF shall enforce the **[**assignment: **TOE's SEE API Access Control SFP]** to provide **[**selection, choose one of: ***restrictive*****]** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/API The TSF shall allow the **[**assignment: **None]** to specify alternative initial values to override the default values when an object or information is created.

FDP_ITC.1 Import of user data without security attributes

FDP_ITC.1.1 The TSF shall enforce the **[**assignment: **Secure Storage Access Control SFP]** when importing SA executables as user data, controlled under the SFP, from outside of the TOE.

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

FDP_ITC.1.3 The TSF shall enforce the following rules when SA executables as user data controlled under the SFP from outside the TOE **[**assignment:

- **The TSF shall decrypt the SA.**
- **The TSF shall verify SA's Mac with the mac from SEE storage.**

]

6.1.4.6 Security Services

FDP_ACC.1/TA **Subset access control**

FDP_ACC.1.1/TA The TSF shall enforce the **[**assignment: **TA Whitelist Access Control SFP]** on

[assignment:

- **Subjects: S.TA_INSTANCE**
- **Objects: OB.SSSA_STORAGE, OB.ATRSA_COUNTER,**
- **Operations:**

  OP.SSSA_CREATE, OP.SSSA_DELETE, OP. SSSA_STORE, OP.SSSA_LOAD,

  OP.CREATE_COUNTER, OP.STEP_COUNTER, OP.EXTRACT_COUNTER.

].

FDP_ACF.1/TA **Security attribute based access control**

FDP_ACF.1.1/TA The TSF shall enforce the **[**assignment: **TA Whitelist Access Control SFP]** to objects based on the following:

[assignment:

- **Subjects (attributes): S.TA_INSTANCE (S.TA_INSTANCE.TA_identity, S.TA_INSTANCE.TEE_identity)**
- **Objects (attributes):**

  OB.SSSA_STORAGE (OB.SSSA_STORAGE.TA_identity, OB.SSSA_STORAGE.TEE_identity)

  OB.ATRSA_COUNTER（OB.ATRSA_COUNTER.TA_identity, OB.ATRSA_COUNTER.TEE_identity）

].

FDP_ACF.1.2/TA The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

[assignment:

- **OP. SSSA_CREATE / OP. SSSA_LOAD / OP. SSSA_STORE / OP. SSSA_DELETE of an object to an OB.SSSA_STORAGE is allowed if the following conditions hold:**

The operation request comes from an instance of the owner of the SecureFlash storage space of Secure Storage SA (S.TA_INSTANCE.TA_identity = OB.SSSA_STORAGE.TA_identity and S.TA_INSTANCE.TEE_identity = OB.SSSA_STORAGE.TEE_identity).

OP.CREATE_COUNTER/ OP.STEP_COUNTER/ OP.EXTRACT_COUNTER of an object to an OB.ATRSA_COUNTER is allowed if the following conditions hold: The operation request comes from an instance of the owner of the monotonic counter of Anti-rollback SA (S.TA_INSTANCE.TA_identity = OB.ATRSA_COUNTER.TA_identity and S.TA_INSTANCE.TEE_identity = OB.ATRSA_COUNTER.TEE_identity).

].

FDP_ACF.1.3/TA The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: **[**assignment: **None]**.

FDP_ACF.1.4/TA The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **[**assignment: **None]**.

FMT_MSA.3/TA **Static attribute initialization**

FMT_MSA.3.1/TA The TSF shall enforce the **[**assignment: **TA Whitelist Access Control SFP]** to provide **[**selection, choose one of: ***restrictive*****]** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/TA The TSF shall allow the **[**assignment: **None]** to specify alternative initial values to override the default values when an object or information is created.

FPT_STM.1/ATRSA **Reliable time stamps**

FPT_STM.1.1/ATRSA The TSF shall be able to provide reliable time stamps.

Application note: The term "reliable time stamps" as used in FPT_STM.1/ATRSA corresponds to a set of monotonic counters, such that the TOE can request to read or increment any counter, but no entity can decrease any of these counters.

FDP_DAU.1/ROTSA **Basic Data Authentication**

FDP_DAU.1.1/ROTSA The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of **[**assignment: **TEE provided certificate]**.

FDP_DAU.1.2/ROTSA The TSF shall provide **[**assignment: **any external entity possessing the root certificate]** with the ability to verify evidence of the validity of the indicated information.

FIA_AFL.1/WSA **Authentication failure handling**

FIA_AFL.1.1/WSA The TSF shall detect when **[**selection: [assignment: ***100***]**]** unsuccessful authentication attempts occur related to [assignment: **Weaver verification**].

FIA_AFL.1.2/WSA When the defined number of unsuccessful authentication attempts has been **[**selection: ***met*****]**, the TSF shall **[**assignment: **prevent further authentication attempts until 60s**].

FIA_UID.1/FESA **Timing of identification**

FIA_UID.1.1/FESA The TSF shall allow [assignment: **query whether the current system authentication is valid**] on behalf of the user to be performed before the user is identified.

FIA_UID.1.2/FESA The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

6.2 Security Assurance Requirements

The development and the evaluation of the TOE shall be done in accordance to the following security assurance requirements: **EAL5 + ALC_DVS.2, and AVA_VAN.5**

The following table shows the assurance requirements by reference the individual components in [CC31R5P3]

Table 6-1 Security Assurance Requirements

Assurance Class	Assurance Components
ASE: Security Target evaluation	ASE_CCL.1: Conformance claims ASE_ECD.1: Extended components definition ASE_INT.1: ST introduction ASE_TSS.1: TOE summary specification ASE_OBJ.2: Security objectives ASE_REQ.2: Derived security requirements ASE_SPD.1: Security problem definition
ALC: Life-cycle support	ALC_CMC.4: Production support, acceptance procedures and automation ALC_CMS.5: Development tools CM coverage ALC_DEL.1: Delivery procedures ALC_DVS.2: Sufficiency of security measures ALC_LCD.1: Developer defined life-cycle model ALC_TAT.2: Compliance with implementation standards
ADV: Development	ADV_ARC.1: Security architecture description ADV_FSP.5: Complete semi-formal functional specification with additional ADV_TDS.4: Semiformal modular design ADV_INT.2: Well-structured internals ADV_IMP.1: Implementation representation of the TSF
AGD: Guidance documents	AGD_OPE.1: Operational user guidance AGD_PRE.1: Preparative procedures
ATE: Tests	ATE_COV.2: Analysis of coverage ATE_DPT.3: Testing: modular design ATE_FUN.1: Functional testing ATE_IND.2: Independent testing - sample
AVA: Vulnerability assessment	AVA_VAN.5: Advanced methodical vulnerability analysis

6.3 Security Requirements Rationale

6.3.1 SFR Necessity and Sufficiency Analysis

Table 6-2 Tracing of security SFR to Security Ojectives

SFR	Security Ojectives
FDP_ITT.1	O.Leak-Inherent O.Leak-Forced
FPT_ITT.1	O.Leak-Inherent O.Leak-Forced O.TOE_Data_Protection
FMT_LIM.1	O.Abuse-Func
FMT_LIM.2	O.Abuse-Func
FAU_SAS.1/Hardware	O.Abuse-Func O.Identification
FDP_SDC.1	O.Phys-Probing
FDP_SDI.2/Hardware	O.Phys-Manipulation O.Leak-Forced
FPT_PHP.3	O.Phys-Probing O.Phys-Manipulation O.Leak-Forced
FRU_FLT.2	O.Malfunction O.Leak-Forced
FPT_FLS.1/Hardware	O.Malfunction O.Leak-Forced
FCS_RNG.1	O.RND
FCS_COP.1/AES	O.Initialization O.Secure_Storage O.SA.Mobile-Root-of-Trust O.Secure_Load_ACode O.Secure_AC_Activation O.Third-Party-SA O.SEE_Isolation O.SA_Isolation O.TOE_Data_Protection
FCS_COP.1/CMAC	O.Initialization O.Secure_Storage

SFR	Security Ojectives
	O.SEE_Isolation O.SA_Isolation O.TOE_Data_Protection O.SA.Mobile-Root-of-Trust
FCS_COP.1/RSA	O.Initialization O.SA.Mobile-Root-of-Trust O.Secure_Load_ACode O.Secure_AC_Activation O.Third-Party-SA
FCS_COP.1/Hash	O.SA.Mobile-Root-of-Trust O.Secure_Load_ACode O.Secure_AC_Activation O.Third-Party-SA O.Initializatioin
FCS_CKM.1/SYM	O.Secure_Storage O.Secure_Load_ACode O.Secure_AC_Activation O.SA.Mobile-Root-of-Trust
FCS_CKM.1/RSA	O.Initialization O.SA.Mobile-Root-of-Trust
FCS_CKM.4	O.Initialization O.Secure_Storage O.SA.Mobile-Root-of-Trust O.Secure_Load_ACode O.Secure_AC_Activation
FAU_SAS.1/ACode	O.Identification
FPT_TUD.1	O.Secure_Load_ACode, O.Third-Party-SA
FPT_INI.1	O.Initialization O.Secure_AC_Activation O.Secure_Storage
FDP_SDI.2/Software	O.Secure_Storage
FTP_ITC.1	O.Secure_Storage O.Initialization O.SA.Secure-Storage O.SA.Anti-Rollback O.SA.Mobile-Root-of-Trust

SFR	Security Ojectives
	O.SA.Weaver O.SA.File-Encryption O.TOE_Data_Protection
FDP_ACC.1/SS	O.Secure_Storage O.Operation O.SA.Secure-Storage O.SEE_Isolation
FDP_ACF.1/SS	O.Secure_Storage O.Operation O.SA.Secure-Storage O.SEE_Isolation
FMT_MSA.3/SS	O.Secure_Storage O.Operation O.SA.Secure-Storage O.SEE_Isolation
FDP_IFC.2/Runtime	O.Operation O.Leak-Inherent O.Leak-Forced O.Runtime_Confidentiality O.SEE_Isolation O.SA_Isolation
FDP_IFF.1/Runtime	O.Operation O.Leak-Inherent O.Leak-Forced O.Runtime_Confidentiality O.SEE_Isolation O.SA_Isolation
FMT_MSA.3/Runtime	O.Operation O.Leak-Inherent O.Leak-Forced O.Runtime_Confidentiality O.SEE_Isolation O.SA_Isolation
FPT_FLS.1/Runtime	O.Operation O.SA_Isolation O.Secure_AC_Activation O.Initialization
FDP_ACC.1/API	O.Operation

SFR	Security Ojectives
FDP_ACF.1/API	O.Operation
FMT_MSA.3/API	O.Operation
FDP_ITC.1	O.Operation
FDP_ACC.1/TA	O.SA.Secure-Storage O.SA.Anti-Rollback
FDP_ACF.1/TA	O.SA.Secure-Storage O.SA.Anti-Rollback
FMT_MSA.3/TA	O.SA.Secure-Storage O.SA.Anti-Rollback
FPT_STM.1/ATRSA	O.SA.Anti-Rollback
FDP_DAU.1/ROTSA	O.SA.Mobile-Root-of-Trust
FIA_AFL.1/WSA	O.SA.Weaver
FIA_UID.1/FESA	O.SA.File-Encryption

Table 6-3 Justification of tracing

Security Objective	Rationale
O.RND	FCS_RNG.1 provide hybrid physical random number generator directly fulfills the objective.
O.Leak-Inherent	FPT_ITT.1, FDP_ITT.1 and FDP_IFC.2/Runtime are suitable to meet the objective.
O.Phys-Probing	The SFR FDP_SDC.1 requires the TSF to protect the confidentiality of the information of the user data stored in specified memory areas and prevent its compromise by physical attacks bypassing the specified interfaces for memory access. The scenario of physical probing as described for this objective is explicitly included in the assignment chosen for the physical tampering scenarios in FPT_PHP.3. Therefore, it is clear that this security functional requirement supports the objective.
O.Malfunction	The definition of this objective shows that it covers a situation, where malfunction of the TOE might be caused by the operating conditions of the TOE (while direct manipulation of the TOE is covered O.Phys-Manipulation). There are two possibilities in this situation: Either the operating conditions are inside the tolerated range or at least one of them is outside of this range. The second case is covered by FPT_FLS.1/Hardware, because it states that a secure state is preserved in this case. The first case is covered by FRU_FLT.2 because it states that the TOE operates correctly under normal (tolerated) conditions.
O.Phys-Manipu	The SFR FDP_SDI.2/Hardware requires the TSF to detect the integrity errors of the stored user data and react in case of detected errors. The

Security Objective	Rationale
lation	scenario of physical manipulation as described for this objective is explicitly included in the assignment chosen for the physical tampering scenarios in FPT_PHP.3. Therefore, it is clear that this security functional requirement supports the objective.
O.Leak-Forced	This objective is directed against attacks, where an attacker wants to force an information leakage, which would not occur under normal conditions. In order to achieve this the attacker has to combine a first attack step, which modifies the behaviour of the TOE (either by exposing it to extreme operating conditions or by directly manipulating it) with a second attack step measuring and analysing some output produced by the TOE. The first step is prevented by the same mechanisms which support O.Malfunction and O.Phys-Manipulation, respectively. The requirements covering O.Leak-Inherent also support O.Leak-Forced because they prevent the attacker from being successful if he tries the second step directly.
O.Abuse-Func	This objective states that abuse of functions must not be possible in Phase 7 of the life-cycle. There are two possibilities to achieve this: (i) They cannot be used by an attacker (i. e. its availability is limited) or (ii) using them would not be of relevant use for an attacker (i. e. its capabilities are limited) since the functions are designed in a specific way. The first possibility is specified by FMT_LIM.2 and the second one by FMT_LIM.1. Since these requirements are combined to support the policy, which is suitable to fulfil O.Abuse-Func, both security functional requirements together are suitable to meet the objective.
O.Identification	Obviously the operations for FAU_SAS.1/Hardware are chosen in a way that they require the TOE to provide the functionality needed for O.Identification. The Initialisation Data (or parts of them) are used for TOE identification. The technical capability of the TOE to store Initialisation Data is provided according to FAU_SAS.1/Hardware. FAU_SAS.1/ACode allows for identification of the Initial TOE.
O.Operation	The following requirements contribute to fulfill the objective: FTP_FLS.1/Runtime states that abnormal operations have to lead to a secure state FDP_ACC.1/SS, FDP_ACF.1/SS, FMT_MSA.3/SS state the policy for controlling access to TOE's storage FDP_IFC.2/Runtime and FDP_IFF.1/Runtime and FMT_MSA.3/Runtime state the policy for controlling access to SA and TOE execution spaces FDP_ACC.1/API and FDP_ACF.1/API and FMT_MSA.3/ API state the policy for controlling access to SEE_API for SA FDP_ITC.1 states the policy for SA loading into TOE.
O.Runtime_Integrity	The following requirements contribute to fulfill the objective: FDP_IFC.2/Runtime and FDP_IFF.1/Runtime and FMT_MSA.3/Runtime state SEE and SA runtime data policy, which grants write access to authorized entities only.
O.Runtime_Co	The following requirements contribute to fulfill the objective

Security Objective	Rationale
nfidentiality	FDP_IFC.2/Runtime and FDP_IFF.1/Runtime and FMT_MSA.3/Runtime state SEE and SA runtime data policy, which grants write access to authorized entities only.
O.SA_Isolation	The following requirements contribute to fulfill the objective: FDP_ACC.1/SS, FDP_ACF.1/ SS, FMT_MSA.3/SS state the policy for controlling access to SA storage FCS_COP.1/AES and FCS_COP.1/CMAC state the cryptographic algorithms used for Secure_Storage to ensure confidentiality of SA data FDP_IFC.2/Runtime and FDP_IFF.1/Runtime and FMT_MSA.3/Runtime state the policy for controlling access to SA execution space FTP_FLS.1/Runtime enforces SA isolation by maintaining a secure state, in particular in case of panic states.
O.Initialization	The following requirements contribute to fulfill the objective: FTP_FLS.1/Runtime states that the TOE has to reach a secure state upon initialization or device binding failure FCS_COP.1/AES, FCS_COP.1/RSA and FCS_COP.1/Hash state the cryptography used to verify the authenticity of TOE firmware FPT_INI.1 enforces the initialization of the TSF through a secure process including the verification of the authenticity and integrity of the TEE firmware.
O.Secure_Load_ACode	The following requirements contribute to fulfill the objective: FCS_COP.1/AES, FCS_COP.1/RSA, FCS_CKM.1/SYM, FCS_CKM.4 and FCS_COP.1/Hash state the cryptography used to verify the authenticity and guarantee the confidentiality of TOE software, which comprises the Additional Code FPT_TUD.1 enforces the update progress of the TSF including the integrity, authenticity and confidentiality of the TOE updated code and data.
O.Secure_AC_Activation	The following requirements contribute to fulfill the objective: FPT_FLS.1/Runtime states that the TOE has to reach a secure state upon initialization or device binding failure FCS_COP.1/AES, FCS_COP.1/RSA, FCS_CKM.1/SYM, FCS_CKM.4 and FCS_COP.1/Hash state the cryptography used to verify the authenticity and guarantee the confidentiality of TOE software, which comprises the Additional Code FPT_INI.1 enforces the initialization of the TSF through a secure process including the verification of the authenticity and integrity of the TEE firmware when the additional code is activated.
O.TOE_Data_Protection	The following requirements contribute to fulfill the objective: FCS_COP.1/AES and FCS_COP.1/CMAC state states the cryptography used to protect consistency and confidentiality of the SEE data in external memory FTP_ITC.1 provide the security protection about some TOE data

Security Objective	Rationale
	FDP_SDI.2/Software monitors the authenticity and consistency of TOE persistent data and states the response upon failure FPT_ITT.1 enforces secure transmission and storage of TOE persistent data.
O.SEE_Isolation	FDP_ACC.1/ SS, FDP_ACF.1/SS, FMT_MSA.3/ SS state the policy for controlling access to TOE storage FCS_COP.1/AES and FCS_COP.1/CMAC state the cryptographic algorithms used for Secure_Storage to ensure confidentiality of TOE data FDP_IFC.2/Runtime, FDP_IFF.1/Runtime and FMT_MSA.3/Runtime state the policy for controlling access to TOE execution space.
O.Third-Party-SA	The following requirements contribute to fulfill the objective: FCS_COP.1/AES, FCS_COP.1/RSA and FCS_COP.1/Hash state the cryptography used to verify the authenticity of the third party SA FPT_TUD.1 enforces the third party SA loading into TOE by security manner.
O.Secure_Storage	The following requirements contribute to fulfill the objective: FCS_COP.1/AES and FCS_COP.1/CMAC states the cryptography used to protect integrity and confidentiality of the SA and SEE data in external memory FTP_ITC.1 provide the security storage to TOE's MAC FDP_ACC.1/SS, FDP_ACF.1/SS, FMT_MSA.3/SS state Storage state the policy for accessing SA and TOE secure storage and protecting the confidentiality of data FDP_SDI.2/Software enforces the consistency and authenticity of the secure storage FDP_SDC.1 enforces the confidentiality of the secure storage FPT_INI.1 enforces the integrity of SEE identification and storage root of trust, and it states the behavior in case of failure FPT_FLS.1 maintains a secure state.
O.SA.Secure-Storage	The following requirements contribute to fulfill the objective: FDP_ACC.1/TA, FDP_ACF.1/TA and FMT_MSA.3/TA state this SA access control about different TA's data using and storage. FDP_ACC.1/SS, FDP_ACF.1/SS, FMT_MSA.3/SS state Storage state the policy for accessing this SA secure storage and protecting the confidentiality of data.
O.SA.Anti-Rollback	The following requirements contribute to fulfill the objective: FDP_ACC.1/TA, FDP_ACF.1/TA and FMT_MSA.3/TA state this SA access control about different TA's counter value. FPT_STM.1/ATRSA states this SA provides a Monotonic-Counter.
O.SA.Mobile-Root-of-Trust	The following requirements contribute to fulfill the objective: FDP_DAU.1/ROTSA states this SA provide TA's certification for TEE

Security Objective	Rationale
	FCS_COP.1/Hash, FCS_COP.1/AES FCS_COP.1/CMAC, FCS_COP.1/RSA, FCS_CKM.1/SYM, FCS_CKM.1/RSA, and FCS_CKM.4 state this SA provides cryptography function and key management.
O.SA.Weaver	The following requirements contribute to fulfill the objective: FIA_AFL.1/WSA states Authentication failure handling.
O.SA.File-Encryption	The following requirements contribute to fulfill the objective: FIA_UID.1/FESA states the TOE shall perform Validity check before this SA is identified.

6.3.2 SFR Dependency Analysis

SFR Missing dependencies justification

Table 6-4 Security requirement dependencies

SFR	Required	Fulfilled by
FDP_ITT.1	FDP_ACC.1 or FDP_IFC.1	FDP_IFC.2/Runtime
FPT_ITT.1	None	Nothing to justify
FMT_LIM.1	FMT_LIM.2	FMT_LIM.2
FMT_LIM.2	FMT_LIM.1	FMT_LIM.1
FAU_SAS.1/Hardware	None	Nothing to justify
FDP_SDC.1	None	Nothing to justify
FDP_SDI.2/Hardware	None	Nothing to justify
FPT_PHP.3	None	Nothing to justify
FRU_FLT.2	FPT_FLS.1	FPT_FLS.1/Hardware
FPT_FLS.1/Hardware	None	Nothing to justify
FCS_RNG.1	None	Nothing to justify
FCS_COP.1/AES	(FCS_CKM.1 or FDP_ITC.1 orFDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/SYM and FCS_CKM.4
FCS_COP.1/CMAC	(FCS_CKM.1 or FDP_ITC.1	FCS_CKM.1/SYM and FCS_CKM.4

SFR	Required	Fulfilled by
	orFDP_ITC.2) and (FCS_CKM.4)	
FCS_COP.1/RSA	(FCS_CKM.1 or FDP_ITC.1 orFDP_ITC.2) and (FCS_CKM.4)	FCS_CKM.1/RSA and FCS_CKM.4
FCS_COP.1/Hash	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2) and (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 cyptographically sensitive parameters which implies that no such parameters must be destroyed. Thus, the dependency on key destruction is unmet.
FCS_CKM.1/SYM	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_COP.1/CMAC, FCS_COP.1/AES and FCS_CKM.4
FCS_CKM.1/RSA	(FCS_CKM.2 or FCS_COP.1) and (FCS_CKM.4)	FCS_COP.1/RSA and FCS_CKM.4
FCS_CKM.4	(FCS_CKM.1 or FDP_ITC.1 or FDP_ITC.2)	FCS_CKM.1/RSA and FCS_CKM.1/SYM
FPT_TUD.1	(FCS_CKM.1 or FCS_CKM.2) and (FCS_COP.1)	FCS_COP.1/RSA, FCS_COP.1/Hash, FCS_COP.1/AES, FCS_CKM.1/SYM and FCS_CKM.1/RSA
FAU_SAS.1/Acode	None	Nothing to justify
FPT_INI.1	None	Nothing to justify
FDP_SDI.2/Software	None	Nothing to justify
FTP_ITC.1	None	Nothing to justify
FDP_ACC.1/SS	FDP_ACF.1	FDP_ACF.1/SS
FDP_ACF.1/SS	FDP_ACC.1 and FMT_MSA.3	FDP_ACC.1/SS and FMT_MSA.3/SS
FMT_MSA.3/SS	FMT_MSA.1 and FMT_SMR.1	The default value cannot be altered or managed, The dependency of FMT_MSA.3 to FMT_MSA.1 and FMT_SMR.1 is unmet.
FDP_IFC.2/Runtime	FDP_IFF.1	FDP_IFF.1/Runtime
FDP_IFF.1/Runtime	FDP_IFC.1 and	FDP_IFC.2/Runtime and FMT_MSA.3/

SFR	Required	Fulfilled by
	FMT_MSA.3	Runtime
FMT_MSA.3/Runtime	FMT_MSA.1 and FMT_SMR.1	The default value cannot be altered or managed, The dependency of FMT_MSA.3 to FMT_MSA.1 and FMT_SMR.1 is unmet.
FPT_FLS.1/Runtime	None	Nothing to justify
FDP_ACC.1/API	FDP_ACF.1	FDP_ACF.1/API
FDP_ACF.1/API	FDP_ACC.1 and FMT_MSA.3	FDP_ACC.1/API and FMT_MSA.3/API
FMT_MSA.3/API	FMT_MSA.1 and FMT_SMR.1	The default value cannot be altered or managed, The dependency of FMT_MSA.3 to FMT_MSA.1 and FMT_SMR.1 is unmet.
FDP_ITC.1	(FDP_ACC.1, FDP_IFC.1) and FMT_MSA.3	FDP_ACC.1/SS and FMT_MSA.3/SS
FDP_ACC.1/TA	FDP_ACF.1	FDP_ACF.1/TA
FDP_ACF.1/TA	FDP_ACC.1 and FMT_MSA.3	FDP_ACC.1/TA and FMT_MSA.3/TA
FMT_MSA.3/TA	FMT_MSA.1 and FMT_SMR.1	The default value cannot be altered or managed, The dependency of FMT_MSA.3 to FMT_MSA.1 and FMT_SMR.1 is unmet.
FPT_STM.1/ATRSA	None	Nothing to justify
FDP_DAU.1/ROTSA	None	Nothing to justify
FIA_AFL.1/WSA	FIA_UAU.1	The Weaver authentication SA supports TAs in implementing the authentication failure handling part of an authentication protocol. The timing of authentication is decided by the TA and hence not by the TSF. Therefore, The dependency of FIA_AFL.1/WSA to FIA_UAU.1 is unmet.
FIA_UID.1/FESA	None	Nothing to justify

6.3.3 SAR Rationale

The Evaluation Assurance Level 5+ has been chosen to commensurate with the threat environment that is experienced by typical consumers of the TOE.

6.3.4 SAR Dependency Analysis

Table 6-5 SAR Missing dependencies justification

SAR	Required	Fulfilled	Missing
ASE_CCL.1	ASE_INT.1, ASE_ECD.1, ASE_REQ.1	ASE_INT.1, ASE_ECD.1, ASE_REQ.1	None
ASE_ECD.1	None	None	None
ASE_INT.1	None	None	None
ASE_TSS.1	ASE_INT.1, ASE_REQ.1, ADV_FSP.1	ASE_INT.1, ASE_REQ.1, ADV_FSP.1	None
ASE_OBJ.2	ASE_SPD.1	ASE_SPD.1	None
ASE_REQ.2	ASE_OBJ.2, ASE_ECD.1	ASE_OBJ.2, ASE_ECD.1	None
ASE_SPD.1	None	None	None
AGD_OPE.1	ADV_FSP.1	ADV_FSP.5(hierarchically above ADV_FSP.1)	None
AGD_PRE.1	None	None	None
ALC_CMC.4	ALC_CMS.1, ALC_DVS.1, ALC_LCD.1	ALC_CMS.1(hierarchically above ALC_CMS.5), ALC_DVS.1(hierarchically above ALC_DVS.2), ALC_LCD.1	None
ALC_CMS.5	None	None	None
ALC_DEL.1	None	None	None
ALC_DVS.2	None	None	None
ALC_LCD.1	None	None	None
ALC_TAT.2	ADV_IMP.1	ADV_IMP.1	
ADV_ARC.1	ADV_FSP.1, ADV_TDS.1	ADV_FSP.5, ADV_TDS.4 (hierarchically above ADV_TDS.1)	None
ADV_FSP.5	ADV_TDS.1,ADV_IMP.1	ADV_TDS.4 (hierarchically above ADV_TDS.1)	None
ADV_TDS.4	ADV_FSP.5	ADV_FSP.5	None

SAR	Required	Fulfilled	Missing
ADV_IMP.1	ADV_TDS.3，ALC_TAT.1	ALC_TAT.2(hierarchically above ALC_TAT.1), ADV_TDS.4(hierarchically above ADV_TDS.3)	None
ADV_INT.2	ADV_IMP.1, ADV_TDS.3, ALC_TAT.1	ADV_IMP.1, ADV_TDS.4(hierarchically above ADV_TDS.3), ALC_TAT.1	None
ATE_COV.2	ADV_FSP.2, ATE_FUN.1	ADV_FSP.5 (hierarchically above ADV_FSP.2), ATE_FUN.1	None
ATE_DPT.3	ADV_ARC.1, ADV_TDS.4, ATE_FUN.1	ADV_ARC.1, ADV_TDS.4, ATE_FUN.1	None
ATE_FUN.1	ATE_COV.1	ATE_COV.2 (hierarchically above ATE_COV.1)	None
ATE_IND.2	ADV_FSP.2, AGD_OPE.1, AGD_PRE.1, ATE_COV.1, ATE_FUN.1	ADV_FSP.5 (hierarchically above ADV_FSP.2), AGD_OPE.1, AGD_PRE.1, ATE_COV.2 (hierarchically above ATE_COV.1), ATE_FUN.1	None
AVA_VAN.5	ADV_ARC.1, ADV_FSP.4, ADV_IMP.1,ADV_TDS.3, AGD_OPE.1, AGD_PRE.1,ATE_DPT.1	ADV_ARC.1, ADV_IMP.1,ADV_FSP.5 (hierarchically above ADV_FSP.4), ADV_TDS.4 (hierarchically above ADV_TDS.3), AGD_OPE.1, AGD_PRE.1	None

7 TOE Summary Specification

7.1 Cryptographic support and random number generation

The implemented physical random number generator together with the associated post processing, and the Random Number Generator can be suitable for generation of signature key pairs, generation of session keys for symmetric encryption mechanisms, random padding bits, generation of seeds for DRNGs.

The TOE also provide AES, SHA, CMAC, RSA cryptographic support. For details about the using of the algorithms, see the following TSS sections.

(FCS_RNG.1, FCS_COP.1/AES, FCS_CKM.1/SYM, FCS_CKM.4, FCS_COP.1/Hash, FCS_COP.1/RSA, FCS_ CKM.1/RSA, FCS_COP.1/CMAC)

7.2 Physical protection

Adopt the high-performance and high-security processor SC300 dedicated to ARM company; SC300 is based on ARM Cortex-M3 frame Architecture, integrating a variety of dedicated processor security features, it has uniform timing，software secure coding making sure branch balanced.

Preventing disturbance attacks: comprising redundant logic and error detection/correction code for the following components Secure core, ROM, RAM, and TOE internal buses, and fault detection logic.

Different module reset mechanisms and self-tests can help against perturbations, and when the frequency is higher or voltage is lower than the timing margin, the critical timing path module will generate system reset.

Preventing side-channel attacks and fault injection attacks: secure core has uniform timing，software secure coding making sure branch balanced、encryption and bus circuit implementation obfuscation、random perturbation、bus has parity checking、CRC..

ROM/RAM/BUS has scrambling, at the same time, physical layout has constrains under shield for providing the protection.

Life cycle control: Turn off debugging permissions based on the LCS. 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 OTP.

Data stored in the TOE are integrity and RAM/ROM adapted data transferred with CRC4/32.

Production data: Data for the identification of the TOE and the associated initialization is stored in the OTP.

（FDP_ITT.1, FPT_ITT.1, FPT_FLS.1 FPT_PHP.3, FDP_SDC.1, FMT_LIM.1, FMT_LIM.2, FRU_FLT.2, FDP_SDI.2/Hardware, FAU_SAS.1/Hardware）

7.3 Security Update and Boot

When the TOE executes the security update progress, the sequence is as following:

1. Load the UpdateOS by security operation.
2. Receive the cipher text update image from out of TOE.
3. The TOE decrypts into a plaintext signed image by AES-256 with platform-derived key and stores it in TOE’s RAM.
4. Verify the plaintext image’s digital signature in RAM by RSA2048-PSS. If this verification fails, stop the execution.
5. Calculate digest for verified image and its certificate by SHA256.
6. Encrypt both the verified image and digest by AES-256 with HUK-derived key, then the encrypted image send to the out of the TOE for storage.

The TOE software version is included in the TOE software image and stored with the image.

The boot image is stored outside the TOE and loaded during startup by the boot loader stored in the TOE. The boot loader performs the following cold boot sequence:

1. Receive the cipher text image from out of TOE.
2. The TOE decrypts into a plaintext signed image and its digest by AES-256 with platform-derived key and stores it in TOE’s RAM.
3. The TOE verifies the digest of the plaintext image and its certificate. If this verification fails, stop the execution.
4. The TOE passes control to the image (jump to Main Control Program).

(FCS_COP.1/RSA, FCS_COP.1/AES, FCS_COP.1/Hash, FPT_TUD.1, FPT_INI.1, FDP_SDI.2/Software, FTP_ITC.1, FDP_ACC.1/SS, FDP_ACF.1/SS, FMT_MSA.3/SS, FAU_SAS.1/ACode).

7.4 TOE's Security Storage

The Secure Flash provides security protection about stored data of SAs and MAC values. It achieves confidential, integrity, and tamper-resistant security protection (this is not in the scope of TOE, it’s assurance by OE.Secure-Component).

The Secure Flash uses hardware communication channels for Connecting to TOE I2C. It takes a communication path with TOE as Master and Secure Flash as Slave. The Master (TOE) must first establish a secure channel with the Secure Flash using the SCP03 protocol, and then the Secure Flash will receive function operation commands from the secure channel and execute them.

The NVM is the non-volatile storage area out of the TOE, and the DDR is the volatile storage area out of the TOE. So TSF data stored out of the TOE are protected the confidentiality, integrity, and tamper-resistant by the following methods:

Confidentiality: The data stored to the NVM / DDR / Secure Flash by the TOE's software is encrypted. The TOE derives the encryption key by HUK using CMAC algorithm (the storage address on NVM / DDR / Secure Flash is used as the input factor). When storing data, encrypt the plaintext by AES128 using the derived key. When reading data, decrypt the cipher text by AES128 using the same key.

Integrity and tamper-resistant: the TOE derived a MAC key by HUK using CMAC algorithm (the storage address on NVM / DDR is used as the input factor). When storing data, the TOE will do AES128-CMAC for the data, the MAC value is stored in Secure Flash, and the MAC value of DDR is stored in TOE's RAM; when reading the data, the TOE will use the MAC value in Secure Flash or RAM for integrity check.

The TOE allocates space based on Secure Flash, NVM, and DDR. The TOE's SEE and each SA have their own dedicated space. The SA management module of the TOE checks the current user in the system. The storage access module of the TOE only provides access to the exclusive space that matches the user's identity.

(FTP_ITC.1, FCS_COP.1/AES, FCS_COP.1/CMAC, FCS_CKM.1/SYM, FCS_COP.1/AES, FDP_ACC.1/SS, FDP_ACF.1/SS, FMT_MSA.3/SS, FDP_SDI.2/Software)

7.5 SA Management

The TOE provides a set of system APIs (different system API functions are distinguished by function numbers) for user-level code or SA call of SEE system.

The SA can call the system API by executing a specific SVC (for example, using SVC #153 as a special entry for SA's system APIs), and specify the number of the system API function to be called with parameters.

Before executing a specified number of system API functions, a whitelist check is performed on the identity of the caller (system or each SA), and the function is executed after the check passes.

The TSA is loaded firstly from out of the TOE, the progress is the same as TOE's update function.

The SA is loaded from the NVM or DDR through the TOE's SA management module, TOE will conduct confidentiality, integrity, and tamper-resistant inspections. When the SA is loaded, the cipher text is decrypted by AES 128. Then the MAC value in TOE's software is used for integrity check by AES128-CMAC.

In the TOE, the SA runs in the application address space (the address space mapped by the MMU). Before running an SA program, the physical RAM where the SA program is located must be mapped to the address space of the MMU through the MMU. TOE uses this MMU mapping mechanism to achieve access isolation protection for the RAM space of each SA.

(FDP_ACC.1/API, FDP_ACF.1/API, FMT_MSA.3/API, FDP_ITC.1, FCS_COP.1/AES, FCS_CKM.1/SYM, FCS_COP.1/CMAC, FDP_IFC.2/Runtime, FDP_IFF.1/Runtime, FMT_MSA.3/Runtime, FPT_TUD.1)

7.6 Security Runtime

TOE's software operating environment distinguishes between privilege level and user level:

- When running in a user-level context, the code at the privileged level cannot be directly executed, it needs to be run through the SVC instruction into the privileged context, and executed in the system call mode.
- Privileged code includes SA driver module and other hardware drivers and other low-level functions that execute at the privileged level and can access hardware resources.

The user-level code includes pure software logic and other middle layers that run on the user level and cannot access hardware resources. When a system call occurs, before the specified system call function is executed at the privilege level, the identity of the caller (system or each SA) is whitelisted and the function is executed after the check is passed.

When the following error occurs, TOE will execute the exception handling mechanism including:

- Illegal access: If TOE system software or SA recognizes that the caller has illegal access, such as insufficient running permissions, attempting to access non-privileged space, and identity authentication does not meet the conditions, etc., then returns error code to the caller.
- Abnormal operation: When TOE's system cannot be repaired by resetting its hardware or software module, the TOE system software sends an interrupt to the outside of the TOE to notify that the TOE needs to be reset, and then the TOE system software pending itself and waiting be reset.
- Failure state: when TOE occurs the device binding failure, operation failure, and initialization failure and so on. The TOE will present a secure state.

(FPT_FLS.1/Runtime，FDP_IFC.2/Runtime，FDP_IFF.1/Runtime，FMT_MSA.3/Runtime)

7.7 Security Service

7.7.1 Secure Storage SA

Secure Storage SA supports up to N data storage spaces allocated on Secure Flash. These N spaces are bounded one by one with the preset TAs (a white list of TA UUIDs in the TOE system for this SA). Each TA can allocate, read, write, and free data in its storage areas on this SA. Different TA's storage data is access control by secure storage SA.

(FDP_ACC.1/TA, FDP_ACF.1/TA, FMT_MSA.3/TA, FDP_ACC.1/SS, FDP_ACF.1/SS, FMT_MSA.3/SS)

7.7.2 Anti-rollback SA

The anti-rollback SA provides a monotonically increasing count value stored securely in Secure Flash to implement the anti-rollback counting function.

The anti-rollback SA supports up to N monotonic counters. The anti-rollback SA function can only be called by the TA preset in the TOE system (a white list of TA UUIDs in the TOE system for this SA). Each TA can allocate, read, and count up its monotonic counter on this SA. Different TA's monotonic counters is access control by anti-rollback SA.

(FPT_STM.1/ATRSA, FDP_ACC.1/TA, FDP_ACF.1/TA, FMT_MSA.3/TA)

7.7.3 Root of Trust SA

The Root of Trust SA provides secure storage that cannot be erased after writing. It is used to save the trusted root certificate, root key. It also provides key management, allowing for key generation, key import and export, and key attestation.

The Root of Trust SA also provides cryptographic functions including:

En/decryption: AES 128, 192, 256 bits by ECB, CBC mode and RSA 2048 bits;

Signature/verification: RSA 2048 bits and SHA 256;

Calculate MAC and verify MAC: CMAC with AES using 128, 192 or 256 bits.

(FCS_RNG.1, FCS_COP.1/AES, FCS_CKM.1/SYM, FCS_CKM.4, FCS_COP.1/Hash, FCS_COP.1/RSA, FCS_ CKM.1/RSA, FCS_COP.1/CMAC, FDP_DAU.1/ROTSA)

7.7.4 Weaver Authentication SA

The Weaver authentication SA provides error penalties:

After 100 consecutive authentication failures, a penalty of 60 seconds before each authentication. The punishment does not end until the authentication is successful.

After authentication success, user can read the value bounded with this authentication password on this SA.

(FIA_AFL.1/WSA)

7.7.5 File Encryption SA

The file encryption SA saves the file encryption key (or key correlate factor) for the encrypted file system.

The File encryption SA provides file encryption key storage function with password/biometric identification protection.

（FIA_UID.1/FESA）

7.7.6 Biometric Authentication Supporting Service for TSA

The TOE limits the Framework APIs that are accessible to the SAs, especially the TSA.

The TOE provides Framework APIs for secure storage of biometric template authentication data and secure storage of Number of retries for failed biometric matching.

(FDP_ACC.1/API, FDP_ACF.1/API, FMT_MSA.3/API, FDP_ACC.1/SS, FDP_ACF.1/SS, FMT_MSA.3/SS)

8 Acronyms

The following table shows the acronyms used in this Security Target

Table 8-1 Acronyms

Acronym	Meaning
ST	Security Target
PP	Protection Profile
CC	Common Criteria
TOE	Target of Evaluation
TSF	TOE Security Functionality
TSFI	TSF Interface
IT	Information Technology
OSP	Organizational Security Policies
EAL	Evaluation Assurance Level
TSC	TSF Scope of Control
TSS	TOE Summary Specification
SA	security application
SEE	Secure Execution Environment
MSP	Mobile Security Processor
TSA	Third party security application
PSA	Platform security application
SOC	System on Chip
OTP	One-Time-Programmable
PERI	peripheral subsystem
NVM	Non-Volatile memory

Acronym	Meaning
DDR	Double Data Rate
NVMC	Non-Volatile memory Controller
CRG	clock and reset generate
SAC	Security Access Controller
IPC	Inter Processor Communication
MBOX	Mail Box
SCE	Symmetric cryptography engine
PKE	Public Key engine
SE_CFG	system config
TRNG	True random number generator
OTA	Over the Air
HUK	Hardware Unique Key
TA	Trusted Application
TEE	Trusted Execution Environment
CA	Client Application
REE	Rich Execution Environment
ATF	Arm Trusted Firmware
MSP	Mobile Security Processor

9 Glossary of Terms

Table 9-1 Glossary of Terms

Term	Meaning
Augmentation	Addition of one or more requirement(s) to a package
Evaluation Assurance Level	Set of assurance requirements drawn from CC Part 3, representing a point on the CC predefined assurance scale, that form an assurance package
Operational Environment	Environment in which the TOE is operated
Protection Profile	Implementation-independent statement of security needs for a TOE type
Security Target	Implementation-dependent statement of security needs for a specific identified TOE
Target Of Evaluation	Set of software, firmware and/or hardware possibly accompanied by guidance

10 Document References

The following table shows the documents referenced in this Security Target

Table 10-1 Document References

Reference	Document
CC31R5P1	Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 5, Part 1: Introduction and general model
CC31R5P2	Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision5, Part 2: Security functional components
CC31R5P3	Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 5, Part 3: Security assurance components
CEM31R5	Common Criteria Evaluation methodology, Version 3.1, Revision 5

