collaborative Protection Profile for USB Portable Devices

DRAFT

Version 0.3

Acknowledgements

This collaborative Protection Profile (cPP) was developed by the USB international Technical Community with representatives from industry, Government agencies, and Common Criteria Test Laboratories.

0. Preface

0.1 Objectives of Document

This document presents the Common Criteria (CC) collaborative Protection Profile (cPP) to express the security functional requirements (SFRs) and security assurance requirements (SARs) for a USB Portable Storage Device. The Evaluation Activities that specify the actions the evaluator performs to determine if a product satisfies the SFRs captured within this cPP are described in <reference to Supporting Document(s)>.

0.2 Scope of Document

The scope of the cPP within the development and evaluation process is described in the Common Criteria for Information Technology Security Evaluation [CC]. In particular, a cPP defines the IT security requirements of a generic type of TOE and specifies the functional and assurance security measures to be offered by that TOE to meet stated requirements [CC1, Section C.1].

0.3 Intended Readership

The target audiences of this cPP are developers, CC consumers, system integrators, evaluators and schemes.

Although the cPPs and SDs may contain editorial errors, cPPs are recognized living documents and the iTCs are dedicated to ongoing updates and revisions. Please report any issues to the USB iTC.

0.4 Related Documents

Common Criteria[1]

[CC1]	Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model, CCMB-2012-09-001, Version 3.1 Revision 4, September 2012.
[CC2]	Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Components, CCMB-2012-09-002, Version 3.1 Revision 4, September 2012.
[CC3]	Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Components, CCMB-2012-09-003, Version 3.1 Revision 4, September 2012.

[1] For details see http://www.commoncriteriaportal.org/

[CEM]	Common Methodology for Information Technology Security Evaluation, Evaluation Methodology, CCMB-2012-09-004, Version 3.1, Revision 4, September 2012.

Other documents

[SD]	Evaluation Activities for USB Portable Storage Device cPP, Version 1.0, DD MONTH 2015

0.5 Revision History

Version	**Date**	**Description**
0.3	2015-07-30	Draft published for Public Review

Contents

Figures / Tables

1. PP Introduction

1.1 PP Reference Identification

PP Reference: collaborative Protection Profile for USB Portable Storage Devices

PP Version: 0.3

PP Date: 2015-07-30

1.2 TOE Overview

This is a collaborative Protection Profile (cPP) whose Target of Evaluation (TOE) is a USB Portable Storage Device. It provides a minimal set of security requirements expected by all USB portable storage devices that target the mitigation of a set of defined threats. This baseline set of requirements will be build upon by future cPPs to provide an overall set of security requirements for USB portable storage devices. A USB Portable Storage Device is a portable storage device (hereafter referred to as “the device” or “the TOE”) that provides a USB interface for connecting to a host computer.

The device employs cryptographic means to provide the necessary protection of user data, and the strength of these cryptographic means lies in the quality of the algorithms, the mode used and the key sizes as well as the entropy of the authorisation factor (e.g., password, passphrase) and cryptographic keys. The device encrypts the user data as it is stored on the device, and decrypts the user data as it leaves the device. All cryptographic functions (encryption/decryption of user data, hashing, random number generation, etc.) are implemented on the device itself. This precludes, for example, encryption/decryption being carried out in a driver on the host computer.

The use of encryption is intended to ensure that data is protected such that even if sophisticated inspection tools were employed to read data from a found or stolen device then any successful access to the data would require a prohibitive cryptanalytic effort.

In addition to protecting the user data, the device is also responsible for ensuring the system data (e.g., software, firmware) cannot be modified by untrusted entities through the logical interface.

The device, its boundaries, components and connections are depicted in Figure 1.

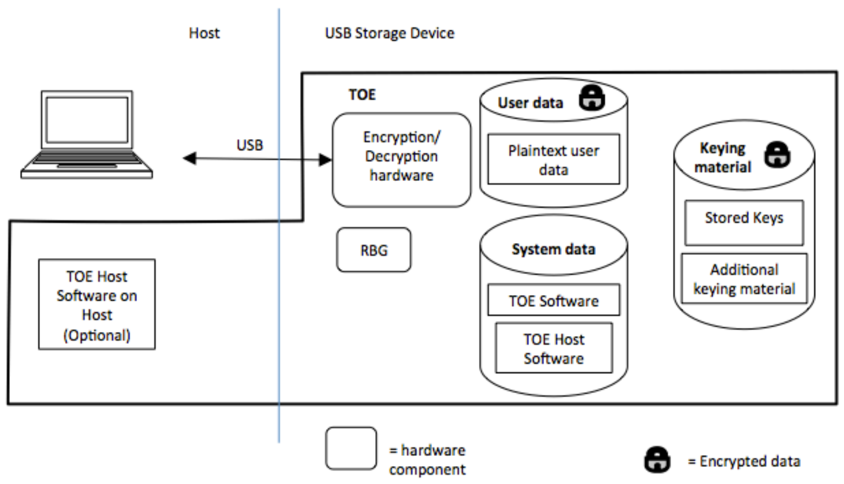


Figure 1: USB device boundaries, components and connections

In this cPP we refer to “authorisation” rather than “authentication” to reflect the fact that the device does not need to hold reference authentication data, or other data that identifies users as individuals. Authorisation data includes passphrases and possibly other data (such as cryptographic salt values), and is used to derive a Key Encryption Key (KEK). The KEK is used to encrypt a Data Encryption Key (DEK) that is generated on the device and used to encrypt and decrypt data stored on the device. To support the intention that a found or stolen device presents a prohibitive cryptanalytic challenge to an attacker, the authorisation data that ultimately give access to the DEK and the user data must not be stored unencrypted on the device.

The host computer plays no role in the encryption/decryption of user data. Software running on the host computer could gather the necessary authorisation data but must not perform any other authorisation functions. The TOE Host software is then considered as part of the TOE and is subject to requirements to provide certain functionality to users of the device.

In addition to user data, the device may also hold data that determines operation of the device itself (including software that may be downloaded to a host computer, such as driver software). This data is referred to as “system data”. System data may include (but are not limited to) software or firmware, patches, or configuration data. The TOE provides the integrity protection of the system data. The host computer plays no role in protecting the system data that resides on the device.

All data on the device is either user data, system data, keying material or else unused (unused areas of the device may contain old encrypted user data or old encrypted keying material).

1.3 TOE Usage

The device is dedicated to storing user data, which may or may not include removable memory media in the device. This cPP is neither suitable for use with more general USB devices that provide access to other forms of storage such as CDs or DVDs, nor for more complex devices that may include memory media (such as smartphone, cameras or media players).

It is intended to be used for the following scenarios:

- Transfer of sensitive data between two host computers.
- Long term storage of sensitive data.

2. CC Conformance

As defined by the references [CC1], [CC2] and [CC3], this cPP:

- conforms to the requirements of Common Criteria v3.1, Revision 4.
- is Part 2 extended, Part 3 conformant
- does not claim conformance to any other PP.

The methodology applied for the PP evaluation is defined in [CEM]. This cPP satisfies the following Assurance Families: APE_CCL.1, APE_ECD.1, APE_INT.1, APE_OBJ.1, APE_REQ.1 and APE_SPD.1.

In order to be conformant to this cPP, a TOE must demonstrate Exact Conformance. Exact Conformance, as a subset of Strict Conformance as defined by the CC, is defined as the ST containing all of the requirements in section 5 (these are the mandatory requirements) of the this cPP, and potentially requirements from Appendix A (these are optional SFRs) or Appendix B (these are selection-based SFRs, some of which will be mandatory according to the selections made in other SFRs) of this cPP. While iteration is allowed, no additional requirements (from the CC parts 2 or 3, or definitions of extended components not already included in this cPP) are allowed to be included in the ST. Further, no requirements in section 5 of this cPP are allowed to be omitted.

3. Security Problem Definition

3.1 Threats

A threat consists of a threat agent, an asset and an adverse action of that threat agent on that asset.

Threat agents are unauthorised users, i.e. they do not possess valid authorisation factors. They are referred to here as “attackers”. Attackers are typically characterised by a number of factors, such as expertise, available resources, and motivation, with motivation being linked directly to the value of the assets at stake. The device is resistant to an attacker possessing a basic attack potential.

The assets to be protected by the device are:

- the plaintext user data and keying material requires confidentiality protection, and
- system data requires integrity protection.

In general, some individual data items stored on the device will not be sensitive, but the cPP does not require distinction of security requirements at this more granular level: all user data and keying material is to be protected for confidentiality, and all system data is to be protected for integrity.

The underlying consequences of all threats considered in this cPP are that an attacker could retrieve plaintext user data or keying material, or could modify system data.

[T.UNAUTHORISED_USER_DATA_ACCESS]

The primary threat to be addressed is the unauthorised disclosure of user data stored on a device. Attackers may attempt to use the logical interface to access plaintext data, or may connect the device to a host that provides raw access to the device content (e.g. to specified disk sectors or blocks).

Attackers may gain physical access to the device, and thus bypass the logical interface to obtain access to user data (perhaps by making physical modifications to the device to access its memory via their own physical connections, or the attacker might use equipment appropriate for the attack potential to read the contents of memory locations from their physical state).

Attackers may also access user data that remains to be unprotected due to a failure interrupting correct operation of the device. Attackers may look for unencrypted keying material giving them unauthorised access to user data.

[T.UNAUTHORISED_SYSTEM_DATA_MODIFICATION]

Attackers may modify system data stored on a device. Attackers may attempt to use the logical interface to modify system data, or may connect the device to a host that provides raw access to the device content (e.g. to specified disk sectors or blocks).

Although the host is not part of the TOE and therefore this Protection Profile does not cover threats to the host, it is noted that TOE objectives to address this threat will also help to protect hosts from attempts to install malware from the TOE's system data.

[T.KEYING_MATERIAL_COMPROMISE]

Possession of any of the keys, authorisation data, random numbers or any other values that contribute to the creation of keys or authorisation data could allow an attacker to defeat the encryption. As part of a conservative approach to security, gaining access to keying material is considered to be of equal importance to gaining access to plaintext user data or system data itself. Attackers may look for keying material in unencrypted sectors of the drive, including in memory used to support power saving modes (in the TOE).

Alternatively an attacker might determine a key because of insufficient entropy used in its generation.

[T.AUTHORISATION_GUESSING]

Attackers may mount an exhaustive search (brute force) attack against the device to determine authorisation factors to gain unauthorised access to the user data stored on the device.

[T.UNAUTHORISED_UPDATE]

Attackers may attempt to perform an unauthorised update of the product, which compromises the security features of the device. Poorly chosen update protocols, signature generation/verification algorithms and parameters may allow attackers to install software and/or firmware that bypasses the intended security features and provides them unauthorised access to user data and keying material or allows modification of system data.

This threat includes attempts to make an unauthorised rollback of updates so that they are no longer applied, or to replay an old, valid update message containing a superseded update (which might allow a known vulnerability to be exploited).

3.2 Assumptions

[A.USER_GUIDANCE]

Users will be instructed in the secure use of the device. This guidance shall include a reminder to generate keys in a suitably secure environment that mitigates the risks of any sort of interference with the key generation process.

[A.LOST_DEVICE]

The device shall be discarded in the event that the device is left unattended, lost or stolen and later recovered, if it may be suspected that an attacker has tampered with the device.

[A.TRUSTED_HOST]

The host computer is trusted, free of malware that could interfere with the correct operation of the device. Only authorised users have access to the host computers.

[A.TRUSTED_CONNECTION]

Communication between host computer and the device is sufficiently protected to prevent disclosure of, or tampering with, user data, keying material or system data.

3.3 Organizational Security Policy

[P.NO_STORE]

It is not possible to reconstruct the keys that protect user data from data persistently stored on the TOE. (This means that complete reference authorisation data is not persistently stored on the TOE.)

[P.RECOVERY]

If the TOE implements any mechanism to enable data recovery in the event of loss of authorisation data[2] then the TOE must allow this mechanism to be disabled, such that it cannot be re-enabled by an unauthorised user.

If no data recovery mechanism is provided by the TOE then this requirement is met. A TOE that provides a data recovery mechanism is required to allow it to be disabled via configuration of the device.

[P.CRYPTO]

[2] Such a mechanism would typically be based on the generation of additional recovery keys that enable the DEK to be recovered via inputs other than the user's usual authorisation data.

The cryptographic algorithms, key lengths and modes used shall be in conformance with the requirements of the National authorised cryptographic authority.

[P.AUTH_CHANGE]

The TOE enables authorised users to change the value of the authorisation data, but only when supplying correct current authorisation data as part of the change operation.

[P.FULL_ENCRYPTION]

Only encrypted data storage shall be available to store user data (i.e. no unencrypted storage is available to users).

[P.NO_BOOT]

It shall not be possible to boot from the TOE in its evaluated configuration.

4. Security Objectives

4.1 Security Objectives for the Operational Environment

[OE.USER_GUIDANCE]

Users will be instructed in the secure use of the device. This guidance shall include a reminder to generate keys in a suitably secure environment that mitigates the risks of any sort of interference with the key generation process.

[OE.LOST_DEVICE]

The device is discarded in the event that the device is left unattended, lost or stolen and later recovered, if it may be suspected that an attacker has tampered with the device.

[OE.TRUSTED_HOST]

The host computer is trusted, free of malware that could interfere with the correct operation of the device. Only authorized users have access to the host computers.

[OE.TRUSTED_CONNECTION]

Communication between host computers and the device is sufficiently protected to prevent disclosure of, or tampering with, user data, keying material or system data.

[OE.NO_BOOT]

The operational environment ensures that the TOE shall not enable itself to be used as a boot device.

5. Security Functional Requirements

The individual security functional requirements are specified in the sections below.

5.1 Conventions

The conventions used in description of the SFRs are as follows:

- Assignments, refinement and selections made by PP author: Indicated with **bold text** and ~~strikethroughs~~, if necessary;
- Assignments and selections that need to be made by the ST writer: Indicated with **bold text**;
- Iteration: Indicated by appending the iteration number in parenthesis, e.g., (a), (b), (c).

Extended SFRs are identified by having a label 'EXT' at the end of the SFR name.

5.2 Class: Cryptographic Support (FCS)

5.2.1 Cryptographic Key Management (FCS_CKM)

5.2.1.1 FCS_CKM.1 Key generation (AES Data Encryption Key)

FCS_CKM.1.1 The TSF shall generate **AES DEK cryptographic keys** ~~in accordance with a specified cryptographic key generation algorithm [assignment: cryptographic key generation algorithm]~~ **using a Random Bit Generator as specified in FCS_RBG_EXT.1** and specified cryptographic key sizes **[selection: 128 bit, 256 bit]** that meet the following: **NIST SP 800-133 (Section 7.1) with ISO 18031 as an approved RBG in addition to those in NIST SP 800-133 (Section 5).**

Application Note 1. : The purpose of this requirement is to explain DEK generation during provisioning. There is no option to import a DEK. The DEK must be generated on the USB device using a suitable random bit generator as specified in FCS_RBG_EXT.1.

5.2.1.2 FCS_CKM.4 Cryptographic key destruction

FCS_CKM.4.1 The TSF shall erase cryptographic keys in accordance with a specified cryptographic key erasure method **[selection:**

- **For volatile memory, the erasure shall be executed by a single direct overwrite [selection: consisting of a pseudo-random pattern using the TSF's RBG, consisting of a pseudo-random pattern using the device's RBG, consisting of zeroes] following by a read-verify.**
- **For non-volatile storage, the erasure shall be executed by:**
  - **A [selection: single, three or more times] overwrite of key data storage location consisting of [selection: a pseudo random pattern using the TSF's**

RBG (as specified in FCS_RBG_EXT.1, a pseudo-random pattern using the device's RBG, a static pattern], followed by a [selection: read-verify, none]. If read-verification of the overwritten data fails, the process shall be repeated again;

] that meets the following: **[selection: NIST SP800-88, no standard].**

Application Note 2. : Keys, including intermediate keys and key material that are no longer needed are destroyed in volatile memory by using one of these approved methods. In these cases, the destruction method conforms to one of methods specified in this requirement. Cryptographic Erase is considered a well defined term for the destruction of key information. Some solutions support write access to media locations where keys are stored, thus allow for destruction of cryptographic keys via direct overwrites of key and key material data. In other cases storage virtualization techniques on system and/or device level could result in multiple copies of key data and/or the underlying media technology does not support direct overwrites of locations where key data are stored. Note that one time programmable memories are excluded.

5.2.2 Cryptographic Operation (FCS_COP)

5.2.2.1 **FCS_COP.1(AES_ENCR) Cryptographic operation (AES User Data Encryption/ Decryption)**

FCS_COP.1.1 The TSF shall perform user data encryption and decryption in accordance with a specified cryptographic algorithm AES used in [selection: CBC, GCM, XTS] mode and cryptographic key sizes [selection: 128 bits, 256 bits] that meet the following: AES as specified in ISO 18033-3, [selection: CBC as specified in ISO 10116, GCM as specified in ISO 19772, and XTS as specified in IEEE 1619].

5.2.3 Cryptographic Key Chaining (FCS_KYC)

5.2.3.1 **FCS_KYC_EXT.1 Key Chaining**

FCS_KYC_EXT.1.1 The TSF shall maintain a chain of intermediary keys originating from the authorisation data and ending in the DEK using the following methods(s): **[selection:**

- **cryptographic authorisation data conditioning as specified in FCS_COP.1(COND),**
- **intermediate key generation as specified in FCS_KDF_EXT.1,**
- **key wrapping as specified in FCS_COP.1(WRAP),**
- **key encryption as specified in FCS_COP.1(ENCR)]**

while maintaining an effective strength of **[selection: 128 bits or 256 bits].**

Application Note 3. : Key chaining is the method of using multiple layers of encryption keys to ultimately secure the protected user data on the TOE. The number of intermediate keys will vary. This applies to all keys that contribute to the ultimate wrapping or derivation of the DEK.

Once the ST Authors have selected a method to create the chain, they pull the appropriate requirement out of Appendix B (Selection-based security requirements). It is allowable for an implementation to use multiple methods.

The method the TOE uses to chain keys and manage/protect them is described in the Key management Description; see Key Management Description for more information.

5.2.4 Salt, Nonce, and Initialization Vector Generation (FCS_SNI)

5.2.4.1 **FCS_SNI_EXT.1 Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation)**

FCS_SNI_EXT.1.1 The TSF shall only use salts that are generated by a RBG in the TSF as specified in FCS_RBG_EXT.1.

FCS_SNI_EXT.1.2 The TSF shall generate and use only unique nonces with a minimum size of 64 bits.

FCS_SNI_EXT.1.3 The TSF shall create IVs in the following manner:

- CBC: IVs shall be non-repeating and unpredictable in accordance with NIST SP 800-38A,
- XTS-AES: No IV. Tweak values shall be **[selection: non-negative integers, assigned consecutively, and starting at an arbitrary non-negative integer, data unit sequence numbers]** and shall satisfy the requirements of IEEE 1619:2007 and may additionally satisfy the requirements of NIST SP 800-38E.
- GCM: IV shall be non-repeating. The number of invocations of GCM shall not exceed 2^32 for a given secret key.

Application Note 4. : This requirement covers several important factors – the salt must be random, but the nonces (for CCM) only have to be unique (a nonce is called a Starting Variable (SV) in ISO/IEC 19772). For tweak values, "assigned consecutively" could mean using a one-up counter.

5.2.5 Random Bit Generation (FCS_RBG)

5.2.5.1 **FCS_RBG_EXT.1 Random Bit Generation**

FCS_RBG_EXT.1.1 The TSF shall perform all deterministic random bit generation services in accordance with **[selection: ISO/IEC 18031:2011, NIST SP 800-90A] using [selection: Hash_DRBG (any), HMAC_DRBG (any), CTR_DRBG (AES)]].**

FCS_RBG_EXT.1.2 The deterministic RBG shall be seeded by an entropy source within the boundary of the device that accumulates entropy from **[selection: a software-based noise source, hardware-based noise source]** with a minimum of **[selection: 128 bits, 256 bits]** of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011 Table C.1 "Security strength table for hash functions", of the keys and hashes that it will generate.

Application Note 5. : ISO/IEC 18031:2011 contains different methods of generating random numbers; each of these, in turn, depends on underlying cryptographic primitives (hash functions/ciphers). The ST author will select the function used and include the specific underlying cryptographic primitives used in the requirement. While any of the identified hash functions (SHA-224, SHA-256, SHA-384, SHA-512) are allowed for Hash_DRBG or HMAC_DRBG, only AES-based implementations for CTR_DRBG are allowed. Table C.2 in ISO/IEC 18031:2011 provides an identification of Security strengths, Entropy and Seed length requirements for the AES-128 and 256 Block Cipher.

The CTR_DRGB in ISO/IEC 18031:2011 requires using derivation function, whereas NIST SP 800-90A does not. Either model is acceptable. In the first selection in FCS_RBG_EXT.1.1, the ST Author choses the standard they are compliant.

The first selection in FCS_RBG_EXT.1.2 allows the ST author to select the type of noise source they employ. It should be noted that a combination of hardware and software based noise sources is acceptable.

It should be noted that the entropy source is considered to be a part of the RBG and as the RBG is included in the TOE, the developer is required to provide the entropy description outlined in Appendix D.

5.2.6 Submask Validation (FCS_VAL)

5.2.6.1 **FCS_VAL_EXT.1 Validation**

FCS_VAL_EXT.1 The TSF shall perform validation of the submask using the following methods: **[selection:**

- **key wrap as specified in FCS_COP.1(WRAP),**
- **hash the authorisation data submask as specified in [selection: FCS_COP.1(HASH), FCS_COP.1(HMAC)] and compare it to a stored hashed value of the authorisation data,**
- **decrypt a known value using the submask as specified in FCS_COP.1(ENCR) and compare it against a stored known value].**

Application Note 6. : The ST Author may need to iterate this requirement if different methods are used to validate the submask.

Once the ST Authors have selected a method to perform validation of the submask, they pull the appropriate requirement out of Appendix B (Selection-based security requirements).

5.3 Class: User Data Protection (FDP)

5.3.1 Protection of User Data on Device (FDP_UDD)

5.3.1.1 **FDP_UDD_EXT.1 Protection of User Data on Device**

FDP_UDD_EXT.1.1 The TSF shall encrypt all user data without user intervention, in accordance with FCS_COP.1(AES_ENCR).

Application Note 7. : The intent of this requirement is to specify that encryption of any protected data will not depend on a user electing to protect that data. The drive encryption specified in FDP_UDD_EXT.1 occurs transparently to the user and the decision to protect the data is outside the discretion of the user, which is a characteristic that distinguishes it from file encryption. The definition of user data can be found in the glossary.

The TOE provides the cryptographic functions to encrypt/decrypt the user data as specified in FCS_COP.1(AES_ENCR).

5.4 Class: Identification and Authentication (FIA)

5.4.1 Authentication Failures (FIA_AFL)

5.4.1.1 **FIA_AFL.1 Authentication failure handling**

FIA_AFL.1.1 The TSF shall detect when **[selection: [assignment: positive integer number], a user configurable positive integer within [assignment: range of acceptable values]]** unsuccessful authentication attempts occur related to **user authentication**.

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been **met**, the TSF shall **delete the current DEK**.

Application Note 8. : The device makes the user data permanently unavailable by deleting the current DEK as soon as a threshold number of consecutive authorisation failures is reached. Authentication attempt is equal to consecutive failed submask validation attempts described in FCS_VAL_EXT.1 (second and third selection options).

Application Note 9. : The developer shall enter the number of unsuccessful authentication attempts or provide a range of acceptable values that could be configured by the authorised user.

5.4.2 Specification of secrets (FIA_SOS)

5.4.2.1 **FIA_SOS.1 Verification of secrets**

FIA_SOS.1.1 The TSF shall provide a mechanism to verify that ~~secrets~~ **authentication data** meet:

- **contains at least 8 characters**

Application Note 10. : The TSF shall reject passphrases that are shorter than 8 characters.

5.4.3 Passphrase support (FIA_PPS)

5.4.3.1 **FIA_PPS_EXT.1 Passphrase support**

FIA_PPS_EXT.1.1 The TSF shall support passphrases that consist of:

- **upper case characters**
- **lower case characters**
- **digits**
- **[assignment: other supported characters]**

FIA_PPS_EXT.1.2 The TSF shall support passphrases with a maximum length of at least 64 characters.

Application Note 11. : The TSF shall support passphrases set by the user that include characters specified in FIA_PPS_EXT.1.1. FIA_PPS_EXT.1.2 specifies the TOE capability to accept passphrases that contain at least 64 characters.

Application Note 12. : The ST author must make the appropriate assignment to specify other characters supported by the TOE.

5.4.4 User authentication (FIA_UAU)

5.4.4.1 **FIA_UAU.6 Re-authenticating**

FIA_UAU.6.1 The TSF shall re-authenticate the user under the conditions:

- **change the value of authorisation data**

Application note 13. : The re-authentication shall use the same authentication mechanism as the initial authentication.

5.4.4.2 **FIA_UAU.7 Protected authentication feedback**

FIA_UAU.7.1 The TSF shall provide only **obscured feedback** to the user while the authentication is in progress.

Application Note 14. : "Obscured feedback" implies that the TSF does not display values of the authorisation data, although an obscured feedback may be provided (such as an asterisk for each character). This requirement is only applicable for devices that allow users to enter the passphrase on the TOE.

5.5 Class: Protection of the TSF (FPT)

5.5.1 Fail secure (FPT_FLS)

5.5.1.1 **FPT_FLS.1 Failure with preservation of secure state**

FPT_FLS.1.1 The TSF shall preserve a secure state when the following types of failures occur:

- **known answer tests self-testing errors during initial start-up**
- **firmware integrity tests errors during initial start-up**

Application Note 15. : If the device fails a self-test, the device shall enter an error state or reset. The error state shall be a "mute" state in which the device does not respond or communicate with the host; in particular, the device shall not decrypt any user data stored on the device and shall not allow modification of the system data.

Application Note 16. : Self-tests include: Known Answer Tests of the cryptographic algorithms supported by the TOE and firmware integrity tests.

5.5.2 TSF self test (FPT_TST)

5.5.2.1 **FPT_TST.1 TSF testing**

FPT_TST.1.1 The TSF shall run a suite of self tests **during initial start-up** to demonstrate the correct operation of **the TSF**.

FPT_TST.1.2 The TSF shall provide authorised users with the capability to verify the integrity of **[selection: [assignment: parts of TSF data], TSF data]**.

FPT_TST.1.3 The TSF shall provide authorised users with the capability to verify the integrity of **[selection: [assignment: parts of TSF], TSF]**.

Application Note 17. : Self-tests include: Known Answer Tests of the cryptographic algorithms supported by the TOE and firmware integrity tests.

5.6 Class: TOE Access (FTA)

5.6.1 TOE access authorisation

5.6.1.1 **FTA_USB_EXT.1 User Authorisation**

FTA_USB_EXT.1.1 The TSF shall require that a valid passphrase is supplied before starting a session that allows any access to user data on the TOE.

FTA_USB_EXT.1.2 The TSF shall authorise the user under the following conditions:

- **connection of the TOE to a host device**
- **recovery of a host device from a power-down or sleep state while the TOE is connected to it**
- **recovery of the TOE from its own power-down or sleep state [assignment: list of other conditions under which re-authorisation is required].**

Authorisation shall require that a valid passphrase is supplied before allowing any access to user data on the TOE.

FTA_USB_EXT.1.3 The TSF shall allow host-initiated termination of the current session.

5.7 Class: Security Management (FMT)

5.7.1 Specification of Management Functions (FMT_SMF)

5.7.1.1 **FMT_SMF.1 Specification of Management Functions**

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions:

- **Disable data recovery mechanism if the device provides such mechanism**
- **Change the value of the authorisation data**
- **Enable data recovery mechanism and then generate the new DEK as specified in FCS_CKM.1**
- **Define a user configurable number of unsuccessful authentication attempts if FIA_AFL.1.1 allows that.**
- **query the current version of the TOE firmware/software**
- **initiate updates to the TOE firmware/software**

Application note 18. : The ST author shall include the fourth bullet in the ST only if the user can configure number of unsuccessful authentication attempts, i.e. the second selection has been made in FIA_AFL.1.1.

Application note 19. : The ST author shall include the last two bullets in the ST only if the TOE provides the capability to update the TOE firmware.

6. Security Assurance Requirements

This cPP identifies the Security Assurance Requirements (SARs) to frame the extent to which the evaluator assesses the documentation applicable for the evaluation and performs independent testing.

This section lists the set of SARs from CC part 3 that are required in evaluations against this cPP. Individual Evaluation Activities to be performed are specified in <reference the SD>.

The general model for evaluation of TOEs against STs written to conform to this cPP is as follows: after the ST has been approved for evaluation, the ITSEF will obtain the TOE, supporting environmental IT (if required), and the administrative/user guides for the TOE. The ITSEF is expected to perform actions mandated by the Common Evaluation Methodology (CEM) for the ASE and ALC SARs. The ITSEF also performs the Evaluation Activities contained within the SD, which are intended to be an interpretation of the other CEM assurance requirements as they apply to the specific technology instantiated in the TOE. The Evaluation Activities that are captured in the SD also provide clarification as to what the developer needs to provide to demonstrate the TOE is compliant with the cPP.

The TOE security assurance requirements are identified in Table 1.

Assurance Class	**Assurance Components**
Security Target (ASE)	Conformance claims (ASE_CCL.1)
	Extended components definition (ASE_ECD.1)
	ST introduction (ASE_INT.1)
	Security objectives for the operational environment (ASE_OBJ.1)
	Stated security requirements (ASE_REQ.1)
	Security Problem Definition (ASE_SPD.1)
	TOE summary specification (ASE_TSS.1)
Development (ADV)	Basic functional specification (ADV_FSP.1)
Guidance documents (AGD)	Operational user guidance (AGD_OPE.1)
	Preparative procedures (AGD_PRE.1)
Life cycle support (ALC)	Labeling of the TOE (ALC_CMC.1)
	TOE CM coverage (ALC_CMS.1)
Tests (ATE)	Independent testing – sample (ATE_IND.1)
Vulnerability assessment (AVA)	Vulnerability survey (AVA_VAN.1)

Table 1: Security Assurance Requirements

6.1 ASE: Security Target

The ST is evaluated as per ASE activities defined in the CEM. In addition, there may be Evaluation Activities specified within the SD that call for necessary descriptions to be included in the TSS that are specific to the TOE technology type.

Appendix D provides a description of the information expected to be provided regarding the quality of entropy in the random bit generator.

Given the criticality of the key management scheme, this cPP requires the developer to provide a detailed description of their key management implementation. This information can be submitted as an appendix to the ST and marked proprietary, as this level of detailed information is not expected to be made publicly available. See Appendix E for details on the expectation of the developer's Key Management Description.

ASE_TSS.1.1C The TOE summary specification shall describe how the TOE meets each SFR**, including supplementary information on Entropy Documentation and Assessment (Appendix D) and a proprietary Key Management Description (Appendix E).**

6.2 ADV: Development

The design information about the TOE is contained in the guidance documentation available to the end user as well as the TSS portion of the ST, and any additional information required by this cPP that is not to be made public (e.g. Entropy Documentation and Assessment) .

6.2.1 Basic Functional Specification (ADV_FSP.1)

The functional specification describes the TOE Security Functions Interfaces (TSFIs). It is not necessary to have a formal or complete specification of these interfaces. Additionally, because TOEs conforming to this PP will necessarily have interfaces to the Operational Environment that are not directly invokable by TOE users, there is little point specifying that such interfaces be described in and of themselves since only indirect testing of such interfaces may be possible. For this cPP, the Evaluation Activities for this family focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation. No additional "functional specification" documentation is necessary to satisfy the Evaluation Activities specified in the SD.

The Evaluation Activities in the SD are associated with the applicable SFRs; since these are directly associated with the SFRs, the tracing in element ADV_FSP.1.2D is implicitly already done and no additional documentation is necessary.

6.3 AGD: Guidance Documentation

The guidance documents will be provided with the ST. Guidance must include a description of how the IT personnel verifies that the Operational Environment can fulfill its role for the security functionality. The documentation should be in an informal style and readable by the IT personnel.

Guidance must be provided for every operational environment that the product supports as claimed in the ST. This guidance includes:

- instructions to successfully install the TSF in that environment; and

- instructions to manage the security of the TSF as a product and as a component of the larger operational environment; and
- instructions to provide a protected administrative capability.

Guidance pertaining to particular security functionality must also be provided; requirements on such guidance are contained in the Evaluation Activities specified in the SD.

6.3.1 Operational User Guidance (AGD_OPE.1)

The operational user guidance does not have to be contained in a single document. Guidance to users can be spread among documents or web pages.

The developer should review the Evaluation Activities contained in the SD to ascertain the specifics of the guidance that the evaluator will be checking for. This will provide the necessary information for the preparation of acceptable guidance. The evaluator is also expected to perform the CEM work units associated with AGD_OPE.1.

6.3.2 Preparative Procedures (AGD_PRE.1)

As with the operational guidance, the developer should look to the Evaluation Activities to determine the required content with respect to preparative procedures. The evaluator is also expected to perform the CEM work units associated with AGD_PRE.1.

6.4 Class ALC: Life-cycle Support

At the assurance level provided for TOEs conformant to this cPP, life-cycle support is limited to end-user-visible aspects of the life-cycle, rather than an examination of the TOE vendor’s development and configuration management process. This is not meant to diminish the critical role that a developer’s practices play in contributing to the overall trustworthiness of a product; rather, it is a reflection on the information to be made available for evaluation at this assurance level.

6.4.1 Labelling of the TOE (ALC_CMC.1)

This component is targeted at identifying the TOE such that it can be distinguished from other products or versions from the same vendor and can be easily specified when being procured by an end user. The evaluator performs the CEM work units associated with ALC_CMC.1.

6.4.2 TOE CM Coverage (ALC_CMS.1)

Given the scope of the TOE and its associated evaluation evidence requirements, the evaluator performs the CEM work units associated with ALC_CMS.1.

6.5 Class ATE: Tests

Testing is specified for functional aspects of the system as well as aspects that take advantage of design or implementation weaknesses. The former is done through the ATE_IND family, while the latter is through the AVA_VAN family. For this cPP, testing is based on advertised functionality and interfaces with dependency on the availability of design information. One of the primary outputs of the evaluation process is the test report as specified in the following requirements.

6.5.1 Independent Testing – Conformance (ATE_IND.1)

Testing is performed to confirm the functionality described in the TSS as well as the operational guidance (includes "evaluated configuration" instructions). The focus of the testing is to confirm that the requirements specified in Section 5 are being met. The Evaluation Activities in the SD identify the specific testing activities necessary to verify compliance with the SFRs. The evaluator produces a test report documenting the plan for and results of testing, as well as coverage arguments focused on the platform/TOE combinations that are claiming conformance to this cPP.

6.6 Class AVA: Vulnerability Assessment

Appendix A in the companion Supporting Document provides a guide to the evaluator in performing a vulnerability analysis.

A. Optional Requirements

As indicated in the introduction to this cPP, the baseline requirements (those that must be performed by the TOE) are contained in the body of this cPP. Additionally, there are two other types of requirements specified in Appendices A and B.

The first type (in this Appendix) is requirements that can be included in the ST, but do not have to be in order for a TOE to claim conformance to this cPP. The second type (in Appendix B) is requirements based on selections in the body of the cPP: if certain selections are made, then additional requirements in that appendix will need to be included in the body of the ST (e.g., cryptographic protocols selected in a trusted channel requirement).

A.1 Cryptographic Support (FCS)

A.1.1 Cryptographic Operation (FCS_COP)

A.1.1.1 FCS_COP.1(SG_VER) Cryptographic Operation (Signature Verification)

FCS_COP.1.1 The TSF shall perform **cryptographic signature services (verification)** in accordance with a specified cryptographic algorithm **[selection:**

- **RSA Digital Signature Algorithm with a key size (modulus) of 2048 bits or greater,**
- **Elliptic Curve Digital Signature Algorithm with a key size of 256 bits or greater]**

~~and cryptographic key sizes [assignment: key sizes]~~ that meet the following: **[selection:**

- **FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS2v1_5; ISO/IEC 9796 -252, Digital signature scheme 2 or Digital Signature scheme 3, for RSA schemes**
- **FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves" P-256, P-384, and [selection: P-521, no other curves]; ISO/IEC 14888-3, Section 6.4, for ECDSA schemes].**

Application Note 20. : This SFR is applicable only if the TOE provides the capability to update the TOE firmware.

Application Note 21. : The ST Author should choose the algorithm implemented to perform digital signatures. For the algorithm(s) chosen, the ST author should make the appropriate assignments/selections to specify the parameters that are implemented for that algorithm.

A.1.1.2 FCS_COP.1(HASH) Cryptographic Operation (Hash Algorithm)

FCS_COP.1.1 The TSF shall perform **cryptographic hashing services** in accordance with a specified cryptographic algorithm **[selection: SHA 256, SHA 512]** ~~and cryptographic~~

~~key sizes [assignment: cryptographic key sizes]~~ that meet the following: **ISO/IEC 10118-3:2004**.

Application Note 22. : This SFR is applicable only if the TOE provides the capability to update the TOE firmware.

A.1.1.3 FCS_COP.1(HMAC) Cryptographic Operation (Keyed hash)

FCS_COP.1.1 The TSF shall perform **keyed hash message authentication** in accordance with a specified cryptographic algorithm **[selection: HMAC-SHA-256, HMAC-SHA-512]** and cryptographic key sizes **[assignment: key size (in bits) used in HMAC]** that meet the following: **ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2".**

Application Note 23. : This SFR is applicable only if the TOE provides the capability to update the TOE firmware.

Application Note 24. : The key size to be used in the HMAC falls into a range between L1 and L2 defined in ISO/IEC 10118 for the appropriate hash function for example for SHA-256 L1 = 512, L2 =256) where L2 ≤ k ≤ L1.

A.1.2 Submask Conditioning (FCS_COP)

A.1.2.1 FCS_SMC_EXT.1 Submask Combining

FCS_SMC_EXT.1.1 The TSF shall combine submasks using the following method **[selection: exclusive OR (XOR), SHA-256, SHA-512]** to generate an intermediary key.

Application Note 25. : This requirement specifies the way that a product may combine the various submasks by using either an XOR or an approved SHA-hash. The approved hash functions are captured in FCS_COP.1(HASH) and FCS_COP.1(HMAC). This requirement should be used only if the TOE combines submasks and will be marked as optional in the cPP.

A.2 Protection of the TSF (FPT)

A.2.1 Trusted Update (FPT_TUD)

A.2.1.1 FPT_TUD_EXT.1 Trusted Update

FPT_TUD_EXT.1.1 The TSF shall provide authorised user the ability to query the current version of the TOE firmware/software.

FPT_TUD_EXT.1.2 The TSF shall provide authorised user the ability to initiate updates to TOE firmware/software.

FPT_TUD_EXT.1.3 The TSF shall verify updates to the system firmware/software updates to the TOE using a **[selection: digital signature mechanism, published hash, keyed hash]** by the manufacture prior to installing those updates.

Application Note 26. : This SFR is applicable only if the TOE provides the capability to update the TOE firmware.

Application Note 27. : The digital signature mechanism referenced in the third element is specified in FCS_COP.1(SG_VER), the published hash mechanism is specified in FCS_COP.1(HASH) and keyed hash mechanism is specified in FCS_COP.1(HMAC).

A.2.2 Trusted Update Rollback (FPT_TUR)

A.2.2.1 FPT_TUR_EXT.1 Trusted Update Rollback

FPT_TUR_EXT.1 The TSF shall not allow rollback of previously applied updates.

Application Note 27.: Prevention of Rollback of previously applied updates is optional even when the device supports updates to the TOE firmware.

B. Selection-Based Requirements

As indicated in the introduction to this PP, the baseline requirements (those that must be performed by the TOE or its underlying platform) are contained in the body of this PP. There are additional requirements based on selections in the body of the PP: if certain selections are made, then additional requirements below will need to be included.

B.1 Cryptographic Operation (FCS_COP)

B.1.1 FCS_COP.1(HASH) Cryptographic Operation (Hash Algorithm)

FCS_COP.1.1 The TSF shall perform **cryptographic hashing services** in accordance with a specified cryptographic algorithm **[selection: SHA 256, SHA 512]** ~~and cryptographic key sizes [assignment: cryptographic key sizes]~~ that meet the following: **ISO/IEC 10118-3:2004**.

Application Note 28. : The hash selection should be consistent with the overall strength of the algorithm used for FCS_KYC_EXT.2. (SHA 256 should be chosen for AES 128-bit keys, SHA 512 should be chosen for AES 256-bit keys).

B.1.2 FCS_COP.1(HMAC) Cryptographic Operation (Keyed hash)

FCS_COP.1.1 The TSF shall perform **keyed hash message authentication** in accordance with a specified cryptographic algorithm **[selection: HMAC-SHA-256, HMAC-SHA-512]** and cryptographic key sizes **[assignment: key size (in bits) used in HMAC]** that meet the following: **ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2".**

Application Note 29. : The key size to be used in the HMAC falls into a range between L1 and L2 defined in ISO/IEC 10118 for the appropriate hash function for example for SHA-256 L1 = 512, L2 =256) where $L2 \le k \le L1$.

B.1.3 FCS_COP.1(COND) Cryptographic Operation (Cryptographic authorisation data conditioning)

FCS_COP.1.1 The TSF shall perform **Password-based Key Derivation Functions** in accordance with a specified cryptographic algorithm **HMAC-[selection: SHA-1, SHA-256, SHA-512]**, **with [assignment: positive integer of 1000 or more] iterations** and **output** cryptographic key sizes **[selection: 128, 256]** that meet the following: **NIST SP 800-132**.

Application Note 30. : Conditioning shall be performed following the process described in NIST SP 800-132. The ST author needs to fill in the number of iterations that are performed. NIAT SP 800-132 also requires the use of a pseudo-random function (PRF) consisting of HMAC with an approved hash function. The ST author selects the hash function used, also includes the appropriate requirements for HMAC.

Application Note 31. The submask, i.e. the output resulting from the conditioning function shall be at least the same length (in number of bits) as the DEK. The ST author makes the appropriate selection.

B.1.4 FCS_COP.1(WRAP) Cryptographic operation (Key Wrapping)

FCS_COP.1.1 The TSF shall perform key wrapping in accordance with a specified cryptographic algorithm AES in the following modes **[selection: KW, KWP, GCM, CCM]** and the cryptographic key size **[selection: 128 bits, 256 bits]** that meet the following: **ISO/IEC 18033-3 (AES), [selection: NIST SP 800-38F, ISO/IEC 19772]**.

Application Note 32. : This requirement shall be used to specify how the DEK is used to wrap the KEK.

Application Note 33. : This requirement may be used in the body of the ST if the ST Author chooses to use key wrapping in the key chaining approach that is specified in FCS_KYC_EXT.2. In such cases, the ST should iterate this SFR, i.e. key wrapping of the DEK and key wrapping of other keys.

B.1.5 FCS_COP.1(ENCR) Cryptographic operation (Key Encryption)

FCS_COP.1.1 The TSF shall perform key encryption and decryption in accordance with a specified cryptographic algorithm AES in the following modes **[selection: CBC, GCM]** and the cryptographic key size **[selection: 128 bits, 256 bits]** that meet the following: **ISO/IEC 18033-3 (AES), [selection: ISO/IEC 10116(CBC), ISO/IEC 19772 (GCM)]**.

Application Note 34. : This requirement is used in the body of the ST if the ST Author chooses to use AES encryption/decryption for protecting they keys as part of the key chaining approach that is specified in FCS_KYC_EXT.2. This requirement will be marked as selection based in the cPP.

B.2 Cryptographic Key Derivation (FCS_KDF)

B.2.1 Cryptographic Key Derivation FCS_KDF_EXT.1

FCS_KDF_EXT.1.1 The TSF shall accept a RNG generated submask as specified in FCS_RBG_EXT.1 to derive an intermediate key, as defined in **[selection: NIST SP 800-108 [selection: KDF in Counter Mode, KDF in Feedback Mode, KDF in Double-Pipeline Iteration Mode], NIST SP 800-132, using the keyed-hash functions specified in FCS_COP.1(HMAC)]**, such that the output is at least of equivalent security strength **[selection: 128 bits or 256 bits]** to the DEK.

Application Note 35. : This requirement is used in the body f the ST if the ST Author chooses to use key derivation in the key chaining approach that is specified in FCS_KYC_EXT.2. This requirement will be marked as selection based in the cPP.

C. Extended Component Definitions

This appendix contains the definitions for the extended requirements that are used in the cPP, including those used in Appendices A and B.

FCS_KDF_EXT.1	Key Derivation Function
FCS_KYC_EXT.1	Key Chaining
FCS_RBG_EXT.1	Random Bit Generation
FCS_SNI_EXT.1	Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation)
FCS_VAL_EXT.1	Validation
FDP_UDD_EXT.1	Protection of User Data on Device
FIA_PPS_EXT.1	Passphrase Support
FPT_TUD_EXT.1	Trusted Update
FPT_TUR_EXT.1	Trusted Update Rollback
FTA_USB_EXT.1	User Authorisation

Table 2 Extended Component Definitions

C.1 Cryptographic Support (FCS)

C.1.1 Cryptographic Key Derivation (FCS_KDF_EXT.1)

Family Behaviour

This family specifies the means by which an intermediate key is derived from a specified set of submasks.

Component levelling

FCS_KDF_EXT.1 Cryptographic Key Derivation
1

FCS_KDF_EXT.1 Cryptographic Key Derivation, requires the TSF to derive intermediate keys from submasks using the specified hash functions.

Management: FCS_KDF_EXT.1

There are no management activities foreseen.

Audit: FCS_KDF_EXT.1

There are no auditable events foreseen.

FCS_KDF_EXT.1 Validation

Hierarchical to: No other components

Dependencies: FCS_COP.1(HMAC) Cryptographic Operation (Keyed hash)

FCS_KDF_EXT.1.1 The TSF shall accept a RNG generated submask as specified in FCS_RBG_EXT.1 to derive an intermediate key, as defined in **[selection: NIST SP 800-108 [selection: KDF in Counter Mode, KDF in Feedback Mode, KDF in Double-Pipeline Iteration Mode], NIST SP 800-132, using the keyed-hash functions specified in FCS_COP.1(HMAC)]**, such that the output is at least of equivalent security strength **[selection: 128 bits or 256 bits]** to the DEK.

Application Note: This requirement is used in the body f the ST if the ST Author chooses to use key derivation in the key chaining approach that is specified in FCS_KYC_EXT.2. This requirement will be marked as selection based in the cPP.

C.1.2 Key Chaining (FCS_KYC_EXT.1)

Family Behaviour

This family provides the specification to be used for using multiple layers of encryption keys to ultimately secure the protected data encrypted on the device.

Component levelling

FCS_KYC_EXT.1 Key Chaining, requires the TSF to maintain a key chain and specifies the characteristics of that chain.

Management: FCS_KYC_EXT.1

There are no management activities foreseen.

Audit: FCS_KYC_EXT.1

There are no auditable events foreseen.

FCS_KYC_EXT.1 Key Chaining

Hierarchical to: No other components

Dependencies: No other components

FCS_KYC_EXT.1 The TSF shall maintain a chain of intermediary keys originating from the authorisation data and ending in the DEK using the following methods(s): **[selection:**

- **cryptographic authorisation data conditioning as specified in FCS_COP.1(COND),**
- **intermediate key generation as specified in FCS_KDF_EXT.1,**
- **key wrapping as specified in FCS_COP.1(WRAP),**
- **key encryption as specified in FCS_COP.1(ENCR)]**

while maintaining an effective strength of **[selection: 128 bits or 256 bits]**.

Application Note: Key chaining is the method of using multiple layers of encryption keys to ultimately secure the protected user data on the TOE. The number of intermediate keys will vary. This applies to all keys that contribute to the ultimate wrapping or derivation of the DEK.

Once the ST Authors have selected a method to create the chain, they pull the appropriate requirement out of Appendix B (Selection-based security requirements). It is allowable for an implementation to use multiple methods.

The method the TOE uses to chain keys and manage/protect them is described in the Key management Description; see Key Management Description (Appendix E) for more information.

C.1.3 Random Bit Generation (FCS_RBG_EXT.1)

Family Behaviour

Components in this family address the requirements for random bit/number generation. This is a new family defined for the FCS class.

Component levelling

FCS_RBG_EXT Random Bit Generation
1

FCS_RBG_EXT.1 Random Bit Generation, requires random bit generation to be performed in accordance with selected standards and seeded by an entropy source.

Management: FCS_RBG_EXT.1

There are no management activities foreseen.

Audit: FCS_RBG_EXT.1

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

Minimal: failure of the randomization process

FCS_RBG_EXT.1 Random Bit Generation

Hierarchical to: No other components

Dependencies: FCS_COP.1(HASH) Cryptographic Operation (Key Hash Algorithm)

FCS_COP.1(HMAC) Cryptographic Operation (Keyed Hash)

FCS_RBG_EXT.1.1 The TSF shall perform all deterministic random bit generation services in accordance with **[selection: ISO/IEC 18031:2011, NIST SP 800-90A]** using **[selection: Hash_DRBG (any), HMAC_DRBG (any), CTR_DRBG (AES)]]**.

FCS_RBG_EXT.1.2 The deterministic RBG shall be seeded by an entropy source within the boundary of the device that accumulates entropy from [selection: a software-based noise source, hardware-based noise source] with a minimum of [selection: 128 bits, 256 bits] of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011 Table C.1 “Security strength table for hash functions”, of the keys and hashes that it will generate.

Application Note: ISO/IEC 18031:2011 contains different methods of generating random numbers; each of these, in turn, depends on underlying cryptographic primitives (hash functions/ciphers). The ST author will select the function used and include the specific underlying cryptographic primitives used in the requirement. While any of the identified hash functions (SHA-224, SHA-256, SHA-384, SHA-512) are allowed for Hash_DRBG or HMAC_DRBG, only AES-based implementations for CTR_DRBG are allowed. Table C.2 in ISO/IEC 18031:2011 provides an identification of Security strengths, Entropy and Seed length requirements for the AES-128 and 256 Block Cipher.

The CTR_DRGB in ISO/IEC 18031:2011 requires using derivation function, whereas NIST SP 800-90A does not. Either model is acceptable. In the first selection in FCS_RBG_EXT.1.1, the ST Author choses the standard they are compliant.

The first selection in FCS_RBG_EXT.1.2 allows the ST author to select the type of noise source they employ. It should be noted that a combination of hardware and software based noise sources is acceptable.

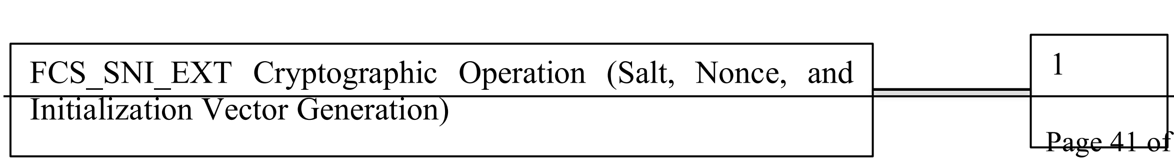
It should be noted that the entropy source is considered to be a part of the RBG and as the RBG is included in the TOE, the developer is required to provide the entropy description outlined in Appendix D.

C.1.4 Key Derivation Function (FCS_SNI_EXT.1)

Family Behaviour

This family ensures that salts, nonces, and IVs are well formed.

Component levelling


FCS_SNI_EXT Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation)
1

FCS_SNI_EXT.1 Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation), requires the generation of salts, nonces, and IVs to be used by the cryptographic components of the TOE to be performed in the specified manner.

Management: FCS_SNI_EXT.1

There are no management activities foreseen.

Audit: FCS_SNI_EXT.1

There are no auditable events foreseen.

FCS_SNI_EXT.1 Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation)

Hierarchical to: No other components

Dependencies: No other components

FCS_SNI_EXT.1.1 The TSF shall only use salts that are generated by a RBG in the TSF as specified in FCS_RBG_EXT.1.

FCS_SNI_EXT.1.2 The TSF shall generate and use only unique nonces with a minimum size of 64 bits.

FCS_SNI_EXT.1.3 The TSF shall create IVs in the following manner:

- CBC: IVs shall be non-repeating and unpredictable in accordance with NIST SP 800-38A,
- XTS-AES: No IV. Tweak values shall be **[selection: non-negative integers, assigned consecutively, and starting at an arbitrary non-negative integer, data unit sequence numbers]** and shall satisfy the requirements of IEEE 1619:2007 and may additionally satisfy the requirements of NIST SP 800-38E.
- GCM: IV shall be non-repeating. The number of invocations of GCM shall not exceed 2^32 for a given secret key.

Application Note: This requirement covers several important factors – the salt must be random, but the nonces (for CCM) only have to be unique (a nonce is called a Starting Variable (SV) in ISO/IEC 19772). For tweak values, "assigned consecutively" could mean using a one-up counter.

C.1.5 Validation (FCS_VAL_EXT.1)

Family Behaviour

This family requires the TSF to provide a method to validate submasks that are provided as an input to a cryptographic function or cryptographic primitive acting as one part of a chain of cryptographic functions that calculates a cryptographic key as the end results of the chain.

Component levelling

FCS_VAL_EXT.1 Validation, requires the TSF to validate submasks by one or more of the specified methods.

Management: FCS_VAL_EXT.1

There are no management activities foreseen.

Audit: FCS_VAL_EXT.1

There are no auditable events foreseen.

FCS_VAL_EXT.1 Validation

Hierarchical to: No other components

Dependencies: FCS_COP.1(WRAP) Cryptographic Operation (Key wrapping)

FCS_COP.1(HASH) Cryptographic Operation (Hash Algorithm)

FCS_COP.1(HMAC) Cryptographic Operation (Keyed Hash)

FCS_VAL_EXT.1 The TSF shall perform validation of the submask using the following methods: **[selection:**

- **key wrap as specified in FCS_COP.1(WRAP),**
- **hash the authorisation data submask as specified in [selection: FCS_COP.1(HASH), FCS_COP.1(HMAC)] and compare it to a stored hashed value of the authorisation data,**
- **decrypt a known value using the submask as specified in FCS_COP.1(ENCR) and compare it against a stored known value].**

C.2 User Data Protection (FDP)

C.2.1 Protection of User Data on Device (FDP_UDD_EXT.1)

Family Behaviour

This family defines requirements on the TSF to protect the user data stored on the device.

Component levelling

FDP_UDD_EXT.1 Protection of User Data on Device, requires the TSF to encrypt all user data stored on the device without user intervention**.**

Management: FDP_UDD _EXT.1

There are no management activities foreseen.

Audit: FDP_UDD_EXT.1

There are no auditable events foreseen.

FDP_UDD_EXT.1 Protection of User Data on Device

Hierarchical to: No other components

Dependencies: FCS_COP.1(AES_ENCR) Cryptographic Operation (AES User Data Encryption/Decryption)

FDP_UDD_EXT.1.1 The TSF shall encrypt all user data without user intervention, in accordance with FCS_COP.1(AES_ENCR).

Application Note: The intent of this requirement is to specify that encryption of any protected data will not depend on a user electing to protect that data. The drive encryption specified in FDP_UDD_EXT.1 occurs transparently to the user and the decision to protect the data is outside the discretion of the user, which is a characteristic that distinguishes it from file encryption. The definition of user data can be found in the glossary.

The TOE provides the cryptographic functions to encrypt/decrypt the user data as specified in FCS_COP.1(AES_ENCR).

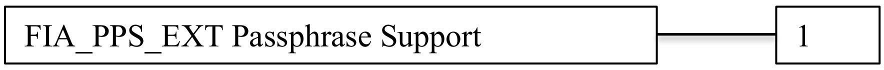
C.3 Identification and Authentication (FIA)

C.3.1 Passphrase Support (FIA_PPS_EXT.1)

Family Behaviour

This family defines requirements on the TSF to support passwords with varying composition requirements, e.g. the minimum length.

Component levelling



FIA_PPS_EXT.1 Passphrase Support, requires the TSF to accept passphrases that consist of the defined characters. It also specifies the minimum maximum length.

Management: FIA_PPS _EXT.1

There are no management activities foreseen.

Audit: FIA_PPS _EXT.1

There are no auditable events foreseen.

FIA_PPS_EXT.1 Passphrase Support

Hierarchical to: No other components

Dependencies: No dependencies

FIA_PPS_EXT.1.1 The TSF shall support passphrases that consist of:

- **upper case characters**
- **lower case characters**
- **digits**
- **[assignment: other supported characters]**

FIA_PPS_EXT.1.2 The TSF shall support passphrases with a maximum length of at least 64 characters.

C.4 Protection of the TSF (FPT)

C.4.1 Trusted Update (FPT_TUD_EXT.1)

Family Behaviour

Components in this family address the requirements for updating the TOE firmware and/or software.

Component levelling

FPT_TUD_EXT.1 Trusted Update, requires the TSF to have the capability to update the TOE firmware and software, including the ability to verify the updated prior to installation.

Management: FPT_TUD _EXT.1

The following actions could be considered for the management functions in FMT:

Ability to update the TOE and to verify the updates.

Audit: FPT_TUD _EXT.1

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

Minimal: Initiation of the update process

Any failure to verify the integrity of the update

FPT_TUD_EXT.1 Trusted Update

FPT_TUD_EXT.1.1 The TSF shall provide authorised user the ability to query the current version of the TOE firmware/software.

FPT_TUD_EXT.1.2 The TSF shall provide authorised user the ability to initiate updates to TOE firmware/software.

FPT_TUD_EXT.1.3 The TSF shall verify updates to the system firmware/software updates to the TOE using a **[selection: digital signature mechanism, published hash, keyed hash]** by the manufacture prior to installing those updates.

C.4.2 Trusted Update Rollback (FPT_TUR_EXT.1)

Family Behaviour

Components in this family address the requirements for allowing the rollback of previous updates to the TOE firmware and/or software.

Component levelling

FPT_TUR_EXT Trusted Update Rollback
1

FPT_TUR_EXT.1 Trusted Update Rollback, requires the TSF to have the capability to rollback previously applied updates to the TOE firmware and/or software.

Management: FPT_TUR_EXT.1

The following actions could be considered for the management functions in FMT:

Ability to update the TOE and to verify the rollback of updates.

Audit: FPT_TUR_EXT.1

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

Minimal: Initiation of the update rollback process

Any failure to verify the integrity of the update

FPT_TUR_EXT.1 Trusted Update Rollback

FPT_TUR_EXT.1 The TSF shall not allow rollback of previously applied updates.

C.5 TOE Access (FTA)

C.5.1 User Authorisation (FTA_USB_EXT.1)

Family Behaviour

This family defines requirements for session establishment and termination.

Component levelling

FTA_USB_EXT.1 User Authorisation, requires the user to present a valid passphrase to establish/initiate a session.

FTA_USB_EXT.1 User Authorisation

FTA_USB_EXT.1.1 The TSF shall require that a valid passphrase is supplied before starting a session that allows any access to user data on the TOE.

FTA_USB_EXT.1.2 The TSF shall authorise the user under the following conditions:

- **connection of the TOE to a host device**
- **recovery of a host device from a power-down or sleep state while the TOE is connected to it**
- **recovery of the TOE from its own power-down or sleep state [assignment: list of other conditions under which re-authorisation is required].**

Authorisation shall require that a valid passphrase is supplied before allowing any access to user data on the TOE.

FTA_USB_EXT.1.3 The TSF shall allow host-initiated termination of the current session.

D. Entropy Documentation And Assessment

This appendix describes the required supplementary information for the entropy source used by the TOE.

The documentation of the entropy source should be detailed enough that, after reading, the evaluator will thoroughly understand the entropy source and why it can be relied upon to provide sufficient entropy. This documentation should include multiple detailed sections: design description, entropy justification, operating conditions, and health testing. This documentation is not required to be part of the TSS in the public facing ST.

D.1 Design Description

Documentation shall include the design of the entropy source as a whole, including the interaction of all entropy source components. Any information that can be shared regarding the design should also be included for any third-party entropy sources that are included in the product.

The documentation will describe the operation of the entropy source to include, how entropy is produced, and how unprocessed (raw) data can be obtained from within the entropy source for testing purposes. The documentation should walk through the entropy source design indicating where the entropy comes from, where the entropy output is passed next, any post-processing of the raw outputs (hash, XOR, etc.), if/where it is stored, and finally, how it is output from the entropy source. Any conditions placed on the process (e.g., blocking) should also be described in the entropy source design. Diagrams and examples are encouraged.

This design must also include a description of the content of the security boundary of the entropy source and a description of how the security boundary ensures that an adversary outside the boundary cannot affect the entropy rate.

If implemented, the design description shall include a description of how third-party applications can add entropy to the RBG. A description of any RBG state saving between power-off and power-on shall be included.

D.2 Entropy Justification

There should be a technical argument for where the unpredictability in the source comes from and why there is confidence in the entropy source delivering sufficient entropy for the uses made of the RBG output (by this particular TOE). . This argument will include a description of the expected min-entropy rate (i.e. the minimum entropy (in bits) per bit or byte of source data) and explain that sufficient entropy is going into the TOE randomizer seeding process. This discussion will be part of a justification for why the entropy source can be relied upon to produce bits with entropy.

The amount of information necessary to justify the expected min-entropy rate depends on the type of entropy source included in the product.

For developer provided entropy sources, in order to justify the min-entropy rate, it is expected that a large number of raw source bits will be collected, statistical tests will be performed, and the min-entropy rate determined from the statistical tests. While no particular statistical tests are required at this time, it is expected that some testing is necessary in order to determine the amount of min-entropy in each output.

For third party provided entropy sources, in which the TOE vendor has limited access to the design and raw entropy data of the source, the documentation will indicate an estimate of the amount of min-entropy obtained from this third-party source. It is acceptable for the vendor to “assume” an amount of min-entropy, however, this assumption must be clearly stated in the documentation provided. In particular, the min-entropy estimate must be specified and the assumption included in the ST.

Regardless of type of entropy source, the justification will also include how the DRBG is initialized with the entropy stated in the ST, for example by verifying that the min-entropy rate is multiplied by the amount of source data used to seed the DRBG or that the rate of entropy expected based on the amount of source data is explicitly stated and compared to the statistical rate. If the amount of source data used to seed the DRBG is not clear or the calculated rate is not explicitly related to the seed, the documentation will not be considered complete.

The entropy justification shall not include any data added from any third-party application or from any state saving between restarts.

D.3 Operating Conditions

The entropy rate may be affected by conditions outside the control of the entropy source itself. For example, voltage, frequency, temperature, and elapsed time after power-on are just a few of the factors that may affect the operation of the entropy source. As such, documentation will also include the range of operating conditions under which the entropy source is expected to generate random data. It will clearly describe the measures that have been taken in the system design to ensure the entropy source continues to operate under those conditions. Similarly, documentation shall describe the conditions under which the entropy source is known to malfunction or become inconsistent - it may be acceptable to simply state that if the device is operating outside of its operating conditions a sufficient level of entropy cannot be guaranteed. Methods used to detect failure or degradation of the source shall be included.

D.4 Health Testing

More specifically, all entropy source health tests and their rationale will be documented. This will include a description of the health tests, the rate and conditions under which each health test is performed (e.g., at startup, continuously, or on-demand), the expected results for each health test, and rationale indicating why each test is believed to be appropriate for detecting one or more failures in the entropy source.

E. Key Management Description

In order to support the evaluation of the SFRs, some additional information is required to describe the keys that implement the SFRs and the ways that these keys relate to each other and to the data that they protect. In particular this information is important as part of a justification that a product Security Target includes an appropriate set of cryptographic SFRs and appropriate selections and assignments within those SFRs. This required supplementary information is referred to in this cPP as the 'Key Management Description', and is specified below.

The Key Management Description should be detailed enough that, after reading, the evaluator will thoroughly understand the keys involved in implementing the SFRs, the data that each key is used to protect and the ways in which each key is used to provide protection. This documentation is not required to be part of the TSS - it can be submitted as a separate document and marked as developer proprietary.

The following topics may not apply to all products, but where a requirement does not apply then the Key Management Description should include an explanation as to why the details do not apply.

The Key Management Description will provide the following information for all keys in the key chain:

- The purpose of the key (indicating its relationship to the Key Reference Model in section 1.3, and the SFRs in the ST that describe the use of the key)
- How and when the key is generated/derived (indicating SFRs in the ST that cover this)
- Whether the key is stored in non-volatile memory
- How and when the key is protected (indicating SFRs in the ST that cover this)
- The strength of the key
- Method of destruction of the key (indicating SFRs in the ST that cover this).

The Key Management Description will also describe the following topics:

- The process for validation of user authorisation shall be described, noting what value(s) is used for validation and the process used to perform the validation. It shall describe how this process ensures no keys in the key chain are weakened or exposed by this process. It shall describe the method used to limit the number of consecutively failed authorisation attempts.
- The authorisation process that leads to the ultimate release of the DEK. This section shall detail the key chain used by the product. It shall describe which keys are used in the protection of the DEK and how they meet the derivation or key wrap requirements. It shall also include any values that add into that key chain or interact with the key chain and the protections that ensure those values do not weaken or expose the overall strength of the key chain.
- The diagram and essay will clearly illustrate the key hierarchy to ensure that at no point could the chain be broken without a cryptographic exhaust or knowledge of the authorisation data, and that the effective strength of the DEK is maintained throughout the Key Chain.

- A description of the data encryption engine, its components, and details about its implementation (e.g. for hardware: integrated within the device's main SOC or separate co-processor, for software: drivers, libraries (if applicable), logical interfaces for encryption/decryption) and identification of any areas that are not encrypted (e.g. partition tables, etc.). The description should also include the data flow from the device's host interface to the device's non-volatile memory storing the data, information on those conditions in which the data bypasses the data encryption engine (e.g. read-write operations to an unencrypted Master Boot Record area). The description should also describe the device's initialisation, the encryption initialisation process, and at what moment the device enables the encryption.
- The process for destroying keys when they are no longer needed by including the type of storage location of all keys and the destruction method for that storage.

The Key Management Description shall include a diagram that provides the following:

- The diagram will include all of keys starting at the conditioning of authorisation data and continuing until the unwrapped DEK is reached. It shall also include any keys or values that contribute to the creation of elements in the chain but are not themselves elements in the chain. It must list the cryptographic strength of each key and illustrate how each key along the chain is protected with either Key Derivation or Key Wrapping (from the allowed options). The diagram should indicate the input used to derive or unwrap each key in the chain.
- A functional (block) diagram showing the main components (such as memories and processors) and the data path between, for hardware, the device's host interface and the device's persistent media storing the data, or for software, the initial steps needed to the activities the TOE performs to ensure it encrypts the storage device entirely when a user or administrator first provisions the product. The hardware encryption diagram shall show the location of the data encryption engine within the data path.

The hardware encryption diagram shall show the location of the data encryption engine within the data path. The evaluator shall validate that the hardware encryption diagram contains enough detail showing the main components within the data path and that it clearly identifies the data encryption engine.

F. Glossary

Term	Meaning
Assurance	Grounds for confidence that a TOE meets the SFRs [CC1].
Data Encryption Key (DEK)	A key used to encrypt data-at-rest.
Error State	The device has failed a self-test and could not reset
Key Chaining	The method of using multiple layers of encryption keys to protect data. A top layer key encrypts a lower layer key which encrypts the data; this method can have any number of layers.
Key Encryption Key (KEK)	A key used to encrypt other keys, such as DEKs or storage that contains keys.
Keying Material	A data item that is used in combination with other data in order to derive a cryptographic key (e.g. a passphrase, seed, or each of the values used in an xor combination).
Passphrase Authorisation Factor	A type of authorisation factor requiring the user to provide a secret set of characters to gain access.
Powered-Off State	The device has been shutdown.
Submask	A submask a bit string that is provided as an input to a cryptographic function or cryptographic primitive acting as one part of a chain of cryptographic functions that calculates a cryptographic key as the end result of the chain. Examples of submasks include: master keys, intermediate keys, wrapping keys, secret bit strings used for authentication or authorisation, and conditioned passphrases.
Target of Evaluation	A set of software, firmware and/or hardware possibly accompanied by guidance. [CC1]
TOE Security Functionality (TSF)	A set consisting of all hardware, software, and firmware of the TOE that must be relied upon for the correct enforcement of the SFRs. [CC1]
TSF Data	Data for the operation of the TSF upon which the enforcement of the requirements relies.

See [CC1] for other Common Criteria abbreviations and terminology.

G. Acronyms

Acronym	Meaning
AES	Advanced Encryption Standard
AF	Authorisation factor
CA	Certificate Authority
CBC	Cipher Block Chaining
CCM	Counter with CBC-Message Authentication Code
cPP	Collaborative protection Profile
DEK	Data Encryption Key
DSA	Digital Signature Algorithm
ECDSA	Elliptic Curve Digital Signature Algorithm
FIPS	Federal Information Processing Standards
GCM	Galois Counter Mode
HMAC	Keyed-Hash Message Authentication Code
IEEE	Institute of Electrical and Electronics Engineers
KDF	Key Derivation Function
KEK	Key Encryption Key
NIST	National Institute of Standards and Technology
MBR	Master Boot Record
PBKDF	Passphrase-Based Key Derivation Function
PP	Protection Profile
RBG	Random Bit Generator
RSA	Rivest Shamir Adleman Algorithm
SHA	Secure Hash Algorithm
SFR	Security Functional Requirement
ST	Security Target
TOE	Target of Evaluation
TSF	TOE Security Functionality
TSS	TOE Summary Specification
USB	Universal Serial Bus
XTS	XEX (XOR Encrypt XOR) Tweakable Block Cipher with Ciphertext Stealing