1  collaborative Protection Profile for Full Drive
2  Encryption – Authorization Acquisition

3  Version 2.0

4  September 09, 2016

# 1 Acknowledgements

2 This collaborative Protection Profile (cPP) was developed by the Full Drive Encryption
3 international Technical Community with representatives from industry, Government
4 agencies, Common Criteria Test Laboratories, and members of academia.

# 1 0. Preface

## 2 0.1 Objectives of Document

3 This document presents the Common Criteria (CC) collaborative Protection Profile (cPP) to
4 express the security functional requirements (SFRs) and security assurance requirements
5 (SARs) for Full Drive Encryption – Authorization Acquisition. The Evaluation Activities that
6 specify the actions the evaluator performs to determine if a product satisfies the SFRs
7 captured within this cPP are described in the *Supporting Document (Mandatory Technical*
8 *Document) Full Drive Encryption: Authorization Acquisition September 2016.*

9 A complete FDE solution requires both an Authorization Acquisition component and
10 Encryption Engine component. A product may provide the entire solution and claim
11 conformance to this cPP (Full Drive Encryption: Authorization Acquisition (FDE-AA)), and
12 the Full Drive Encryption: Encryption Engine (FDE-EE) cPP.

13 However, because the FDE-AA/EE Protection Profile suite is in its infancy, it is not yet
14 possible to mandate that all dependent products will conform to a cPP. Non-validated
15 dependent products (i.e., EE) may be considered to be an acceptable part of the Operational
16 Environment for the AA TOE/product on a case-by-case basis as determined by the relevant
17 national scheme.

18 The FDE iTC intends to develop guidance for developers whose products provide both
19 components (i.e., an AA and EE) to aid them in developing a Security Target (ST) that can
20 claim conformance to both FDE cPPs. One important aspect to note is:

21 **Note to ST authors**: **There is a selection in the ASE_TSS that must be completed. One**
22 **cannot simply reference the SARs in this cPP.**

## 23 0.2 Scope of Document

24 The scope of the cPP within the development and evaluation process is described in the
25 Common Criteria for Information Technology Security Evaluation. In particular, a cPP
26 defines the IT security requirements of a technology specific type of TOE and specifies the
27 functional and assurance security requirements to be met by a compliant TOE.

## 28 0.3 Intended Readership

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

1 ## 0.4 Related Documents

2 **Protection Profiles**

3 [FDE – EE]   collaborative Protection Profile for Full Drive Encryption – Encryption
4 Engine, Version 2.0, September 09, 2016

5 **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.

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

    [SD]         Supporting Document (Mandatory Technical Document), Full Drive
                    Encryption: Authorization Acquisition September 2016

6

---

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

1 ## 0.5 Revision History

| Version | Date | Description |
|---------|------|-------------|
| 0.1 | August 26, 2014 | Initial Release for iTC review |
| 0.2 | September 5, 2014 | Draft published for Public review |
| 0.14 | October 17, 2014 | Incorporated comments received from the Public review |
| 1.0 | January 26, 2015 | Incorporated comments from CCDB review |
| 1.5 | September 22, 2015 | Revised based on additional use cases developed by iTC |
| 2.0 | September 09, 2016 | Incorporated comments received from the public review, and also updated the Key Destruction section and AVA_VAN. |

2

# 1 Contents

2

# Figures / Tables

# 1. PP Introduction

## 1.1    PP Reference Identification

PP Reference: collaborative Protection Profile for Full Drive Encryption – Authorization Acquisition

PP Version: 2.0

PP Date:  September 09, 2016

## 1.2    Introduction to the FDE Collaborative Protection Profiles (cPPs) Effort

The purpose of the first set of Collaborative Protection Profiles (cPPs) for *Full Drive Encryption (FDE): Authorization Acquisition (AA)* and *Encryption Engine (EE)* is to provide requirements for Data-at-Rest protection for a lost device that contains storage. These cPPs allow FDE solutions based in software and/or hardware to meet the requirements. The form factor for a storage device may vary, but could include: system hard drives/solid state drives in servers, workstations, laptops, mobile devices, tablets, and external media. A hardware solution could be a Self-Encrypting Drive or other hardware-based solutions; the interface (USB, SATA, etc.) used to connect the storage device to the host machine is outside the scope of this cPP.

Full Drive Encryption encrypts all data (with certain exceptions) on the storage device and permits access to the data only after successful authorization to the FDE solution. The exceptions include the necessity to leave a portion of the storage device (the size may vary based on implementation) unencrypted for such things as the Master Boot Record (MBR) or other AA/EE pre-authentication software. These FDE cPPs interpret the term "full drive encryption" to allow FDE solutions to leave a portion of the storage device unencrypted so long as it contains plaintext user or plaintext authorization data.

Since the FDE cPPs support a variety of solutions, two cPPs describe the requirements for the FDE components shown in Figure 1.



*Figure 1: FDE components*

The *FDE cPP - Authorization Acquisition* describes the requirements for the Authorization Acquisition piece and details the security requirements and assurance activities necessary to interact with a user and result in the availability of sending a Border Encryption Value (BEV) to the Encryption Engine.

1 The *FDE cPP - Encryption Engine* describes the requirements for the Encryption Engine
2 piece and details the necessary security requirements and assurance activities for the actual
3 encryption/decryption of the data by the DEK. Each cPP will also have a set of core
4 requirements for management functions, proper handling of cryptographic keys, updates
5 performed in a trusted manner, audit and self-tests.

6 This TOE description defines the scope and functionality of the Authorization Acquisition,
7 and the Security Problem Definition describes the assumptions made about the operating
8 environment and the threats to the AA that the cPP requirements address.

## 9    1.3    Implementations

10 Full Drive Encryption solutions vary with implementation and vendor combinations.

11 Therefore, vendors will evaluate products that provide both components of the Full Disk
12 Encryption Solution (AA and EE) against both cPPs – could be done in a single evaluation
13 with one ST. A vendor that provides a single component of a FDE solution would only
14 evaluate against the applicable cPP. The FDE cPP is divided into two documents to allow
15 labs to independently evaluate solutions tailored to one cPP or the other. When a customer
16 acquires an FDE solution, they will either obtain a single vendor product that meets the AA +
17 EE cPPs or two products, one of which meets the AA and the other of which meets the EE
18 cPPs.

19 The table below illustrates a few *examples* for certification.

20                                       *Table 1: Examples of cPP Implementations*

| Implementation | cPP | Description |
|---|---|---|
| Host | AA | Host software provides the interface to a self-encrypting drive |
| Self-Encrypting Drive (SED) | EE | A self-encrypting drive used in combination with separate host software |
| Software FDE | AA + EE | A software full drive encryption solution |
| Hybrid | AA + EE | A single vendor's combination of hardware (e.g. hardware encryption engine, cryptographic co-processor) and software |

## 21    1.4    Target of Evaluation (TOE) Overview

22 The Target of Evaluation (TOE) for this cPP (Authorization Acquisition) may be either a
23 Host software solution that manages a HW Encryption Engine (e.g. a SED) or as part of a
24 combined evaluation of this cPP and the Encryption Engine cPP for a vendor that is
25 providing a solution that includes both components.

1 The following sections provide an overview of the functionality of the FDE AA as well as the
2 security capabilities.

### 1.4.1 Authorization Acquisition Introduction

4 The Authorization Acquisition sends a Border Encryption Value (BEV), which could be a
5 Key Encryption Key (KEK), a Key Releasing Key (KRK), or some other type of key to the
6 Encryption Engine. The EE does not have to use this value directly as the key to decrypt or
7 release the DEK. It may use it as part of a scheme that uses other intermediate keys to
8 eventually protect the DEK. A KEK wraps other keys, notably the DEK or other intermediary
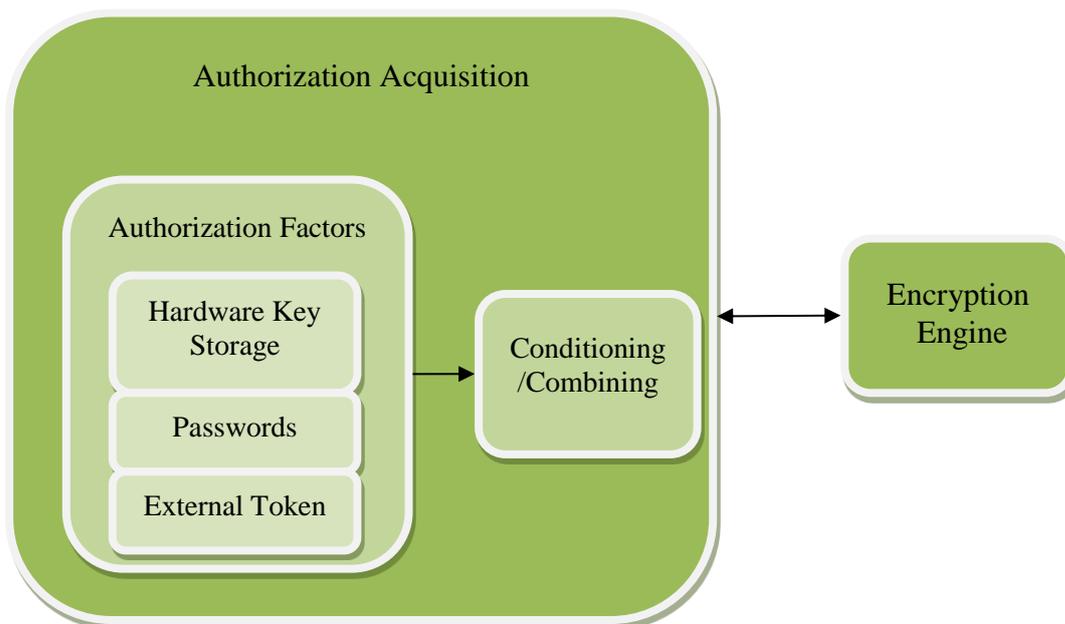


*Figure 2: Authorization Acquisition Details*

9 keys which chain to the DEK. Key Releasing Keys (KRKs) authorize the EE to release either
10 the DEK or other intermediary keys which chain to the DEK. Figure 2 illustrates the
11 components within AA and its relationship with EE.

12 Authorization factors may be unique to individual users or may be used by a group of
13 individuals. In other words, the EE requires authorization factors from the AA to establish
14 that the possessor of the authorization factor belongs to the community of users authorized to
15 access information stored on the storage device (and does not require specific user
16 authorization). Examples of authorization factors include, but are not limited to, passwords,
17 passphrases, or randomly generated values stored on USB tokens or a pin to release a key on
18 hardware storage media such as a Trusted Platform Module (TPM).

1  **1.4.2  Authorization Acquisition Security Capabilities**

2  The AA collects authorization factors which the EE uses to access data on the storage device
3  and perform a variety of management functions. Depending on the type of authorization
4  factor, the AA may condition them further. For example, it may apply an approved password-
5  based key derivation function (e.g. PBKDF2) on passwords. An external token containing a
6  randomly generated value of sufficient strength may require no further conditioning on the
7  authorization factors. The AA may then combine one or more authorization factors in such a
8  way that maintains the strength of both factors.

9  The AA serves as the main management interface to the EE. However, the EE may also offer
10 management functionality. The requirements in the EE cPP address how the EE should
11 handle these features. The management functionality may include the ability to send
12 commands to the EE such as changing a DEK, setting up new users, managing KEKs and
13 other intermediate keys, and performing a key sanitization (e.g. overwrite of the DEK). It
14 may also forward commands that partition the drive for use by multiple users. However, this
15 document defers the management of partitions and assumes that administrators and users will
16 only provision and manage the data on whole drives.

17 **1.4.3  Interface/Boundary**

18 The interface and boundary between the AA and the EE will vary based on the
19 implementation. If one vendor provides the entire FDE solution, then it is may choose to not
20 implement an interface between the AA and EE components. If a vendor provides a solution
21 for one of the components, then the assumptions below state that the channel between the two
22 components is sufficiently secure. Although standards and specifications exist for the
23 interface between AA and EE components, the cPP does not require vendors to follow the
24 standards in this version.

25 **1.5    The TOE and the Operational/Pre-Boot Environments**

26 The environment in which the AA functions may differ depending on the boot stage of the
27 platform in which it operates, see Figure 3. Depending on the solution's archiecture, aspects
28 of provisioning, initialization, and authorization may be performed in the Pre-Boot
29 environment, while encryption, decryption and management functionality are likely
30 performed in the Operating System environment. In non-software solutions,
31 encryption/decryption starts in Pre-OS environment and continues into OS present
32 environment.

33 In the Operating System environment, the Authorization Acquisition has the full range of
34 services available from the operating system (OS), including hardware drivers, cryptographic
35 libraries, and perhaps other services external to the TOE.

36 The Pre-Boot environment is much more constrained with limited capabilities. This
37 environment turns on the minimum number of peripherals and loads only those drivers
38 necessary to bring the platform from a cold start to executing a fully functional operating
39 system with running applications.

1  The AA TOE may include or leverage features and functions within the operational
2  environment.
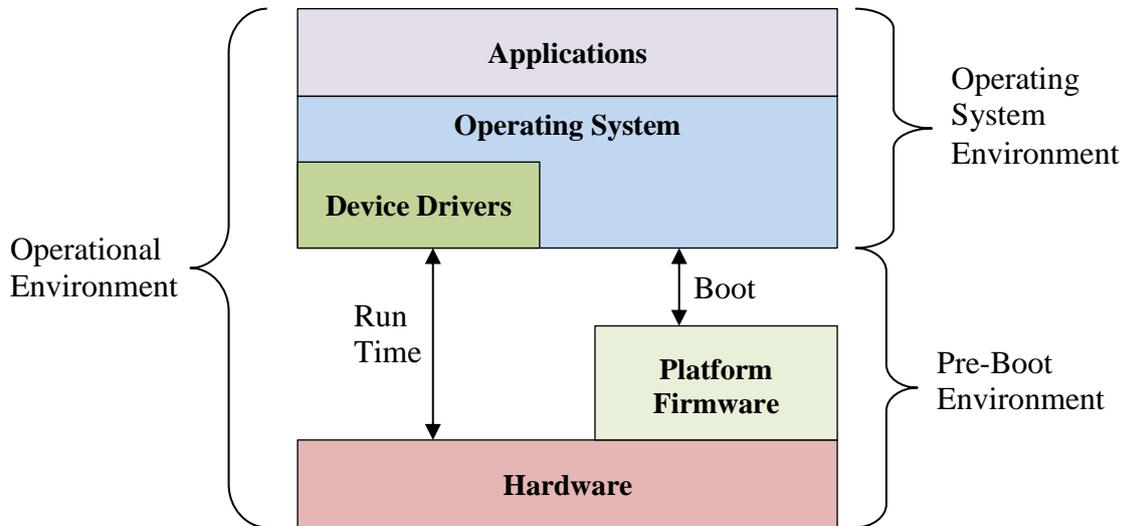


*Figure 3: Operational Environment*

3

## 1.6   TOE Use Case

5  The use case for a product conforming to the FDE cPPs is to protect data at rest on a device
6  that is lost or stolen while powered off without any prior access by an adversary. The use case
7  where an adversary obtains a device that is in a powered state and is able to make
8  modifications to the environment or the TOE itself (e.g., evil maid attacks) is not addressed
9  by these cPPs (i.e., FDE-AA and FDE- EE).

# 2. CC Conformance Claims

As defined by the references [CC1], [CC2], and [CC3], this cPP conforms to the requirements of Common Criteria v3.1, Release 4. This cPP is conformant to CC v3.1, r4, CC Part 2 and CC Part 3 conformant. Extended component definitions can be found in Appendix C.

The methodology applied for the cPP 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.

This cPP does not claim conformance to another PP.

In order to be conformant to this cPP, a TOE must demonstrate *Exact Conformance*. *Exact Conformance* is defined as the ST containing all of the requirements in section 5 of this cPP, and potentially requirements from Appendix A or Appendix B of this cPP. While iteration is allowed, no additional requirements (from the CC parts 2 or 3) are allowed to be included in the ST. Further, no requirements in section 5 of this cPP are allowed to be omitted.

# 1 3. Security Problem Definition

## 2 3.1 Threats

3 This section provides a narrative that describes how the requirements mitigate the mapped
4 threats. A requirement may mitigate aspects of multiple threats. A requirement may only
5 mitigate a threat in a limited way. Some requirements are optional, either because the TSF
6 fully mitigates the threat without the additional requirement(s) being claimed or because the
7 TSF relies on its Operational Environment to provide the functionality that is described by
8 the optional requirement(s).

9 A threat consists of a threat agent, an asset and an adverse action of that threat agent on that
10 asset. The threat agents are the entities that put the assets at risk if an adversary obtains a lost
11 or stolen storage device. Threats drive the functional requirements for the target of evaluation
12 (TOE). For instance, one threat below is T.UNAUTHORIZED_DATA_ACCESS. The threat
13 agent is the possessor (unauthorized user) of a lost or stolen storage device. The asset is the
14 data on the storage device, while the adverse action is to attempt to obtain those data from the
15 storage device. This threat drives the functional requirements for the storage device
16 encryption (TOE) to authorize who can use the TOE to access the hard disk and
17 encrypt/decrypt the data. Since possession of the KEK, DEK, intermediate keys,
18 authorization factors, submasks, and random numbers or any other values that contribute to
19 the creation of keys or authorization factors could allow an unauthorized user to defeat the
20 encryption, this SPD considers key material equivalent to the data in importance and they
21 appear among the other assets addressed below.

22 It is important to reemphasize at this point that this collaborative Protection Profile does not
23 expect the product (TOE) to defend against the possessor of the lost or stolen hard disk who
24 can introduce malicious code or exploitable hardware components into the Target of
25 Evaluation (TOE) or the Operational Environment. It assumes that the user physically
26 protects the TOE and that the Operational Environment provides sufficient protection against
27 logical attacks. One specific area where a conformant TOE offers some protection is in
28 providing updates to the TOE; other than this area, though, this cPP mandates no other
29 countermeasures. Similarly, these requirements do not address the "lost and found" hard disk
30 problem, where an adversary may have taken the hard disk, compromised the unencrypted
31 portions of the boot device (e.g., MBR, boot partition), and then made it available to be
32 recovered by the original user so that they would execute the compromised code.

1    (T.UNAUTHORIZED_DATA_ACCESS) The cPP addresses the primary threat of
2    unauthorized disclosure of protected data stored on a storage device. If an adversary obtains a
3    lost or stolen storage device (e.g., a storage device contained in a laptop or a portable external
4    storage device), they may attempt to connect a targeted storage device to a host of which they
5    have complete control and have raw access to the storage device (e.g., to specified disk
6    sectors, to specified blocks).

7    [FCS_AFA_EXT.2,      FMT_MOF.1,      FMT_SMF.1,      FPT_PWR_EXT.1,
8    FPT_PWR_EXT.2, FCS_VAL_EXT.1, FPT_TST_EXT.1]

9    Rationale: [FCS_AFA_EXT.2] requires authentication to be re-entered upon return
10    from a compliant power state. [FMT_MOF.1] restricts the ability to modify compliant
11    power states to administrators. [FPT_PWR_EXT.1] defines what power states are
12    compliant for the TOE. [FPT_PWR_EXT.2] defines conditions in which the TOE will
13    enter a compliant power state. These requirements ensure the device is secure if lost in
14    a compliant power state.

15    [FMT_SMF.1] ensures the TSF provides the functions necessary to manage important
16    aspects of the TOE including requests to change and erase the DEK. The correct
17    behaviour of all cryptographic functionality is verified through the use of self-tests
18    [FPT_TST_EXT.1]. [FCS_VAL_EXT.1] verifies correct authentication and limits
19    attempts to decrypt the data.

20    (T.KEYING_MATERIAL_COMPROMISE) Possession of any of the keys, authorization
21    factors, submasks, and random numbers or any other values that contribute to the creation of
22    keys or authorization factors could allow an unauthorized user to defeat the encryption. The
23    cPP considers possession of key material of equal importance to the data itself. Threat agents
24    may look for key material in unencrypted sectors of the storage device and on other
25    peripherals in the operating environment (OE), e.g. BIOS configuration, SPI flash.

26    [FCS_AFA_EXT.1, FCS_AFA_EXT.2, FCS_CKM.4(a), FCS_CKM.4(b),
27    FCS_CKM_EXT.4(a), FCS_CKM_EXT.4(b), FCS_KYC_EXT.1, FMT_MOF.1,
28    FMT_SMF_.1, FCS_KYP_EXT.1, FPT_PWR_EXT.1, FPT_PWR_EXT.2,
29    FCS_SNI_EXT.1, FCS_VAL_EXT.1, FPT_TST_EXT.1, FCS_CKM.1(a),
30    FCS_CKM.1(b), FCS_COP.1(b), FCS_COP.1(c), FCS_COP.1(d), FCS_COP.1(e),
31    FCS_COP.1(f), FCS_COP.1(g), FCS_KDF_EXT.1, FCS_PCC_EXT.1,
32    FCS_RBG_EXT.1, FCS_SMC_EXT.1]

33    Rationale: The keying material that threat agents may attempt to compromise are
34    generated as specified by [FCS_CKM.1(a) and (b)], both of which are generated
35    properly via [FCS_RBG_EXT.1]. One or more submasks [FCS_AFA_EXT.1] may be
36    combined [FCS_SMC_EXT.1] and/or chained [FCS_KYC_EXT.1] to produce the
37    BEV. The key chain can be maintained by several methods, including:

38    • Key derivation [FCS_KDF_EXT.1]
39    • Key wrapping [FCS_COP.1(d)]

- Key combining [FCS_SMC_EXT.1]
- Key transport [FCS_COP.1(e)]
- Key encryption [FCS_COP.1(g)]

These requirements ensure the BEV is properly generated and protected. Proper generation of salts, nonces, and IVs [FCS_SNI_EXT.1] ensures conditioning of authentication factors and to support cryptographic functions requiring their use (such as symmetric key generation and AES encryption and decryption using Galois/Counter Mode [GCM]). FCS_VAL_EXT.1 defines methods for validation of keying material such as hashing [FCS_COP.1(b)], keyed-hash message authentication [FCS_COP.1(c)], and decrypting a known value with the keying material [FCS_COP.1(f)]. Key data can also be protected using submask combining [FCS_SMC_EXT.1] which can also be done using a hash function. The correct behavior of all cryptographic functionality is verified through the use of self-tests [FPT_TST_EXT.1].

FPT_KYP_EXT.1 ensures unwrapped key material is not stored in non-volatile memory and [FCS_CKM_EXT.4(a)] along with [FCS_CKM.4(a)] ensures proper key material destruction; minimizing the exposure of plaintext keys and key material.

Secure power management is essential to ensuring that power saving states cannot be used by an attacker to access plaintext keying material. The TSF defines Compliant power saving states [FPT_PWR_EXT.1] that encrypt or destroy [FCS_CKM.4(b), FCS_CKM_EXT.4(b)] all keying material when entered by various conditions [FPT_PWR_EXT.2]. This material is not decrypted until a valid authorization factor is provided [FCS_AFA_EXT.2]. FMT_MOF.1 restricts the ability to modify compliant power states to administrators.

FMT_SMF.1 ensures the TSF provides the functions necessary to manage important aspects of the TOE including generating and configuring authorization factors and the power saving states that the TSF uses.

(T.AUTHORIZATION_GUESSING) Threat agents may exercise host software to repeatedly guess authorization factors, such as passwords and PINs. Successful guessing of the authorization factors may cause the TOE to release BEV or otherwise put it in a state in which it discloses protected data to unauthorized users.

[FCS_AFA_EXT.1, FCS_SNI_EXT.1, FCS_PCC_EXT.1, FCS_SMC_EXT.1
FCS_VAL_EXT.1]

Rationale: [FCS_VAL_EXT.1] requires several options for enforcing validation, such as key sanitization of the DEK or when a configurable number of failed validation attempts is reached within a 24 hour period. This mitigates brute force attacks against authorization factors such as passwords and pins.

[FCS_AFA_EXT.1] requires a set of authorization factors that will be difficult to guess, user provides factors are conditioned [FCS_PCC_EXT.1] to increase the cost of repeatedly guessing a user provided value.  [FCS_SNI_EXT.1] requires proper salts,

which will prevent pre-computed attacks. FCS_SMC_EXT.1 allows for multifactor authentication options, further increasing the difficulty of guessing a authentication factor.

(T.KEYSPACE_EXHAUST) Threat agents may perform a cryptographic exhaust against the key space. Poorly chosen encryption algorithms and/or parameters allow attackers to exhaust the key space through brute force and give them unauthorized access to the data.

[FCS_KYC_EXT.1, FCS_CKM.1(a), FCS_CKM.1(b), FCS_RBG_EXT.1]

Rationale: [FCS_CKM.1(a) and (b)] and [FCS_RBG_EXT.1] ensure cryptographic keys are random and of an appropriate strength/length to make exhaustion attempts cryptographically difficult and cost prohibitive. [FCS_KYC_EXT.1] ensures all keys protecting the BEV are of the same strength.

(T.UNAUTHORIZED_UPDATE) Threat agents may attempt to perform an update of the product which compromises the security features of the TOE. Poorly chosen update protocols, signature generation and verification algorithms, and parameters may allow attackers to install software and/or firmware that bypasses the intended security features and provides them unauthorized access to data.

[FCS_COP.1(a) (optional), FMT_SMF.1, FPT_TUD_EXT.1]

Rationale: FPT_TUD_EXT.1 provides authorized users the ability to query the current version of the TOE software/firmware, initiate updates, and verify updates prior to installation using a manufacturer digital signature. FCS_COP.1(a) defines the signature function that is used to verify updates.

FMT_SMF.1 ensures the TSF provides the functions necessary to manage important behavior of the TOE which includes the initiation of system firmware/software updates.

## 3.2    Assumptions

Assumptions that must remain true in order to mitigate the threats appear below:

(A.INITIAL_DRIVE_STATE) Users enable Full Drive Encryption on a newly provisioned or initialized storage device free of protected data in areas not targeted for encryption. The cPP does not intend to include requirements to find all the areas on storage devices that potentially contain protected data. In some cases, it may not be possible - for example, data contained in "bad" sectors.

While inadvertent exposure to data contained in bad sectors or un-partitioned space is unlikely, one may use forensics tools to recover data from such areas of the storage device. Consequently, the cPP assumes bad sectors, un-partitioned space, and areas that must contain unencrypted code (e.g., MBR and AA/EE pre-authentication software) contain no protected data.

1       [OE.INITIAL_DRIVE_STATE]

2   (A.SECURE_STATE) Upon the completion of proper provisioning, the drive is only
3   assumed secure when in a powered off state up until it is powered on and receives initial
4   authorization.

5       [OE.POWER_DOWN]

6   (A.TRUSTED_CHANNEL) Communication among and between product components (e.g.,
7   AA and EE) is sufficiently protected to prevent information disclosure. In cases in which a
8   single product fulfils both cPPs, then the communication between the components does not
9   extend beyond the boundary of the TOE (e.g., communication path is within the TOE
10  boundary). In cases in which independent products satisfy the requirements of the AA and
11  EE, the physically close proximity of the two products during their operation means that the
12  threat agent has very little opportunity to interpose itself in the channel between the two
13  without the user noticing and taking appropriate actions.

14      [OE.TRUSTED_CHANNEL]

15  (A.TRAINED_USER) Authorized users follow all provided user guidance, including keeping
16  password/passphrases and external tokens securely stored separately from the storage device
17  and/or platform.

18      [OE.PASSPHRASE_STRENGTH, OE.POWER_DOWN, OE.SINGLE_USE_ET,
19      OE.TRAINED_USERS]

20  (A.PLATFORM_STATE) The platform in which the storage device resides (or an external
21  storage device is connected) is free of malware that could interfere with the correct operation
22  of the product.

23      [OE.PLATFORM_STATE]

24  (A.SINGLE_USE_ET) External tokens that contain authorization factors are used for no
25  other purpose than to store the external token authorization factors.

26      [OE.SINGLE_USE_ET]

27  (A.POWER_DOWN) The user does not leave the platform and/or storage device unattended
28  until all volatile memory is cleared after a power-off, so memory remnant attacks are
29  infeasible.

30  Authorized users do not leave the platform and/or storage device in a mode where sensitive
31  information persists in non-volatile storage (e.g., lock screen). Users power the platform
32  and/or storage device down or place it into a power managed state, such as a "hibernation
33  mode".

34      [OE.POWER_DOWN]

1  (A.PASSWORD_STRENGTH) Authorized administrators ensure password/passphrase
2  authorization factors have sufficient strength and entropy to reflect the sensitivity of the data
3  being protected.

4      [OE.PASSPHRASE_STRENGTH]

5  (A.PLATFORM_I&A) The product does not interfere with or change the normal platform
6  identification and authentication functionality such as the operating system login. It may
7  provide authorization factors to the operating system's login interface, but it will not change
8  or degrade the functionality of the actual interface.

9      [OE.PLATFORM_I&A]

10  (A.STRONG_CRYPTO) All cryptography implemented in the Operational Environment and
11  used by the product meets the requirements listed in the cPP. This includes generation of
12  external token authorization factors by a RBG.

13      [OE.STRONG_ENVIRONMENT_CRYPTO]

14  (A.PHYSICAL) The platform is assumed to be physically protected in its Operational
15  Environment and not subject to physical attacks that compromise the security and/or interfere
16  with the platform's correct operation.

17      [OE.PHYSICAL]

## 18  3.3    Organizational Security Policy

19  There are no organizational security policies addressed by this cPP.

# 1   4. Security Objectives

## 2   4.1   Security Objectives for the Operational Environment

3 The Operational Environment of the TOE implements technical and procedural measures to
4 assist the TOE in correctly providing its security functionality. This part wise solution forms
5 the security objectives for the Operational Environment and consists of a set of statements
6 describing the goals that the Operational Environment should achieve.

7 (OE.TRUSTED_CHANNEL) Communication among and between product components (i.e.,
8 AA and EE) is sufficiently protected to prevent information disclosure.

9       Rationale: In situations where there is an opportunity for an adversary to interpose
10      themselves in the channel between the AA and the EE a trusted channel must be
11      established to prevent exploitation. [A.TRUSTED_CHANNEL] assumes the existence
12      of a trusted channel between the AA and EE, except for when the boundary is within
13      and does not breach the TOE or is in such close proximity that a breach is not possible
14      without detection.

15 (OE.INITIAL_DRIVE_STATE) The OE provides a newly provisioned or initialized storage
16 device free of protected data in areas not targeted for encryption.

17      Rationale: Since the cPP requires all protected data to encrypted A.
18      INITIAL_DRIVE_STATE assumes that the initial state of the device targeted for FDE
19      is free of protected data in those areas of the drive where encryption will not be invoked
20      (e.g., MBR and AA/EE pre-authentication software). Given this known start state, the
21      product (once installed and operational) ensures partitions of logical blocks of user
22      accessible data is protected.

23 (OE.PASSPHRASE_STRENGTH) An authorized administrator will be responsible for
24 ensuring that the passphrase authorization factor conforms to guidance from the Enterprise
25 using the TOE.

26      Rationale: Users are properly trained [A.TRAINED_USER] to create authorization
27      factors that conform to administrative guidance.

28 (OE.POWER_DOWN) Volatile memory is cleared after power-off so memory remnant
29 attacks are infeasible.

30      Rationale: Users are properly trained [A.TRAINED_USER] to not leave the storage
31      device unattended until powered down or placed in a managed power state such as
32      "hibernation mode". A.POWER_DOWN stipulates that such memory remnant attacks
33      are infeasible given the device is in a powered-down or "hibernation mode" state.

34  (OE.SINGLE_USE_ET) External tokens that contain authorization factors will be used for
35 no other purpose than to store the external token authorization factor.

1            Rationale: Users are properly trained [A.TRAINED_USER] to use external token
2            authorization factors as intended and for no other purpose.

3 (OE.STRONG_ENVIRONMENT_CRYPTO) The Operating Environment will provide a
4 cryptographic function capability that is commensurate with the requirements and capabilities
5 of the TOE and Appendix A.

6            Rationale: All cryptography implemented in the Operational Environment and used by
7            the product meets the requirements listed in this cPP [A.STRONG_CRYPTO].

8 (OE.TRAINED_USERS) Authorized users will be properly trained and follow all guidance
9 for securing the TOE and authorization factors.

10            Rationale: Users are properly trained [A.TRAINED_USER] to create authorization
11            factors that conform to guidance, not store external token authorization factors with
12            the device, and power down the TOE when required (OE.PLATFORM_STATE) The
13            platform in which the storage device resides (or an external storage device is
14            connected) is free of malware that could interfere with the correct operation of the
15            product.

16            A platform free of malware [A.PLATFORM_STATE] prevents an attack vector that
17            could potentially interfere with the correct operation of the product.

18 (OE.PLATFORM_STATE) The platform in which the storage device resides (or an external
19 storage device is connected) is free of malware that could interfere with the correct operation
20 of the product.

21            Rationale: A platform free of malware [A.PLATFORM_STATE] prevents an attack
22            vector that could potentially interfere with the correct operation of the product.

23 (OE.PLATFORM_I&A) The Operational Environment will provide individual user
24 identification and authentication mechanisms that operate independently of the authorization
25 factors used by the TOE.

26            Rationale: While the product may provide authorization factors to the Operating
27            system's login interface, it must not change or degrade the functionality of the actual
28            interface. A.PLATFORM_I&A requires that the product not interfere or change the
29            normal platform I&A functionality.

30 (OE.PHYSICAL) The Operational Environment will provide a secure physical computing
31 space such than an adversary is not able to make modifications to the environment or to the
32 TOE itself.

33            Rationale: As stated in section 1.6, the use case for this cPP is to protect data at rest on a
34            device where the adversary receives it in a powered off state and has no prior access.

35

# 5. Security Functional Requirements

The individual security functional requirements are specified in the sections below. Based on selections made in these SFRs it will also be necessary to include some of the selection-based SFRs in Appendix B. Additional optional SFRs may also be adopted from those listed in Appendix A for those functions that are provided by the TOE instead of its Operational Environment.

The Evaluation Activities defined in [SD] describe actions that the evaluator will take in order to determine compliance of a particular TOE with the SFRs. The content of these Evaluation Activities will therefore provide more insight into deliverables required from TOE Developers.

## 5.1    Conventions

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

- Assignment: Indicated with *italicized text*;

- Refinement made by PP author: Indicated with **bold text** or ~~strikethroughs~~ for text that is added to or removed from the original SFR;

- Selection: Indicated with <u>underlined text</u>;

- Assignment within a Selection: Indicated with <u>*italicized and underlined text*</u>;

- Iteration: Indicated by appending the SFR with parentheses that contain a letter that is unique for each iteration, e.g. (a), (b), (c) and/or with a slash (/) followed by a descriptive string for the SFR's purpose, e.g. /Server.

SFR text that is bold, italicized, and underlined indicates that the original SFR defined an assignment operation but the PP author completed that assignment by redefining it as a selection operation, which is also considered to be a refinement of the original SFR.

If the selection or assignment is to be completed by the ST author, it is preceded by 'selection:' or 'assignment:'. If the selection or assignment has been completed by the PP author and the ST author does not have the ability to modify it, the proper formatting convention is applied but the preceding word is not included. The exception to this is if the SFR definition includes multiple options in a selection or assignment and the PP has excluded certain options but at least two remain. In this case, the selection or assignment operations that are not permitted by this PP were removed without applying additional formatting and the 'selection:' or 'assignment:' text is preserved to show that the ST author still has the ability to choose from the reduced set of options.

Extended SFRs (i.e. those SFRs that are not defined in CC Part 2) are identified by having a label '_EXT' at the end of the SFR name.

1    ## 5.2    SFR Architecture

2    The following table lists the SFRs that are mandated by this cPP.

3    *Table 2: TOE Security Functional Requirements*

| Functional Class | Functional Components |
|---|---|
| Cryptographic Support (FCS) | FCS_AFA_EXT.1 Authorization Factor Acquisition |
|  | FCS_AFA_EXT.2 Timing of Authorization Factor Acquisition |
|  | FCS_CKM.4(a) Cryptographic Key Destruction (Power Management) |
|  | FCS_CKM.4(d) Cryptographic Key Destruction (Software TOE, 3rd Party Storage) |
|  | FCS_CKM_EXT.4(a) Cryptographic Key and Key Material Destruction (Destruction Timing) |
|  | FCS_CKM_EXT.4(b) Cryptographic Key and Key Material Destruction (Power Management) |
|  | FCS_KYC_EXT.1 Key Chaining (Initiator) |
|  | FCS_SNI_EXT.1 Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) |
| Security Management (FMT) | FMT_MOF.1 Management of Functions Behavior |
|  | FMT_SMF.1 Specification of Management Functions |
| Protection of the TSF (FPT) | FPT_KYP_EXT.1 Protection of Key and Key Material |
|  | FPT_PWR_EXT.1 Power Saving States |
|  | FPT_PWR_EXT.2 Timing of Power Saving States |
|  | FPT_TUD_EXT.1 Trusted Update |

4    ## 5.3    Class: Cryptographic Support (FCS)

5    *FCS_AFA_EXT.1 Authorization Factor Acquisition*

6    **FCS_AFA_EXT.1.1** The TSF shall accept the following authorization factors: [selection:

7    - a submask derived from a password authorization factor conditioned as defined in
8      FCS_PCC_EXT.1,
9    - an external Smartcard factor that is at least the same bit-length as the DEK, and is
10     protecting a submask that is [selection: generated by the TOE (using the RBG as
11     specified in FCS_RBG_EXT.1), generated by the Host Platform] protected using
12     RSA with key size [selection: 2048 bits, 3072 bits, 4096 bits], with user presence
13     proved by presentation of the smartcard and [selection: none, an OE defined PIN,
14     a configurable PIN].
15   - an external USB token factor that is at least the same security strength as the BEV,
16     and is providing a submask generated by the TOE, using the RBG as specified in
17     FCS_RBG_EXT.1,
18   - an external USB token factor that is at least the same security strength as the BEV,
19     and is providing a submask generated by the Host Platform

20   ].

21   *Application Note: This requirement specifies what authorization factors the TOE accepts*
22   *from the user.  A password entered by the user is one authorization factor that the TOE must*

*be able to condition, as specified in FCS_PCC_EXT.1. Another option is a smart card authorization factor, with the differentiating feature being how the value is generated – either by the TOE's RBG or by the platform. An external USB token may also be used, with the submask value generated either by the TOE's RBG or by the platform.*

*The TOE may accept any number of authorization factors, and these are categorized as "submasks". The ST author selects the authorization factors they support, and there may be multiple methods for a selection.*

*Use of multiple authorization factors is preferable; if more than one authorization factor is used, the submasks produced must be combined using FCS_SMC_EXT.1 specified in Appendix A.*

**FCS_AFA_EXT.2 Timing of Authorization Factor Acquisition**

**FCS_AFA_EXT.2.1** The TSF shall reacquire the authorization factor(s) specified in FCS_AFA_EXT.1 upon transition from any Compliant power saving state specified in FPT_PWR_EXT.1 prior to permitting access to plaintext data.

*Application Note: This should be accomplished by clearing keys that are no longer needed so that keys must be derived or decrypted again.*

**FCS_CKM.4(a) Cryptographic Key Destruction (Power Management)**

**FCS_CKM.4.1(a)** The TSF shall [selection: instruct the Operational Environment to clear, erase] cryptographic keys and key material from volatile memory when transitioning to a Compliant power saving state as defined by FPT_PWR_EXT.1 that meets the following: [*a key destruction method specified in FCS_CKM.4(d)*].

*Application Note: In some cases, erasure of keys from volatile memory is only supported by the Operational Environment, in which case the Operational Environment must expose a well-documented mechanism or interface to invoke the memory clearing operation.*

**FCS_CKM.4(d) Cryptographic Key Destruction (Software TOE, 3rd Party Storage)**

**FCS_CKM.4.1(d) Refinement:** The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method [***selection:***

- ***For volatile memory, the destruction shall be executed by a [selection:***
    - ○ ***single overwrite consisting of [selection:***
        - ▪ ***a pseudo-random pattern using the TSF's RBG,***
        - ▪ ***zeroes,***
        - ▪ ***ones,***
        - ▪ ***a new value of a key,***
        - ▪ ***[assignment: some value that does not contain any CSP]],***
    - ○ ***removal of power to the memory,***

    o  ***destruction of reference to the key directly followed by a request for garbage collection];***
- ***For non-volatile storage that consists of the invocation of an interface provided by the underlying platform that [selection:***
  - ***logically addresses the storage location of the key and performs a [selection: single, [assignment: ST author defined multi-pass]] overwrite consisting of [selection:***
    - ***a pseudo-random pattern using the TSF's RBG,***
    - ***zeroes,***
    - ***ones,***
    - ***a new value of a key,***
    - ***[assignment: some value that does not contain any CSP];***
  - ***instructs the underlying platform to destroy the abstraction that represents the key]***

]

that meets the following: [*no standard*].

*Application Note: This SFR is FCS_CKM.4(d), to align with the numbering in the FDE EE cPP.*

*The interface referenced in the requirement could take different forms, the most likely of which is an application programming interface to an OS kernel. There may be various levels of abstraction visible. For instance, in a given implementation the application may have access to the file system details and may be able to logically address specific memory locations. In another implementation the application may simply have a handle to a resource and can only ask the platform to delete the resource. The level of detail to which the TOE has access will be reflected in the TSS section of the ST.*

*Several selections allow assignment of a 'value that does not contain any CSP'. This means that the TOE uses some other specified data not drawn from an RBG meeting FCS_RBG_EXT requirements, and not being any of the particular values listed as other selection options. The point of the phrase 'does not contain any CSP' is to ensure that the overwritten data is carefully selected, and not taken from a general 'pool' that might contain current or residual data that itself requires confidentiality protection.*

*FCS_CKM_EXT.4(a) Cryptographic Key and Key Material Destruction (Destruction Timing)*

**FCS_CKM_EXT.4.1(a)** The TSF shall destroy all keys and key material when no longer needed.

*Application Note: Keys, including intermediate keys and key material that are no longer needed are destroyed by using an approved method, FCS_CKM.4(d). Examples of keys are intermediate keys, submasks, and BEV. There may be instances where keys or key material that are contained in persistent storage are no longer needed and require destruction. Based*

*on their implementation, vendors will explain when certain keys are no longer needed. There are multiple situations in which key material is no longer necessary, for example, a wrapped key may need to be destroyed when a password is changed. However, there are instances when keys are allowed to remain in memory, for example, a device identification key. If a PIN was used for a SmartCard, ensuring that the PIN was properly destroyed shall be addressed.*

*FCS_CKM_EXT.4(b) Cryptographic Key and Key Material Destruction (Power Management)*

**FCS_CKM_EXT.4.1(b) Refinement:** The TSF shall destroy all **key material, BEV, and authentication factors stored in plaintext** when **transitioning to a Compliant power saving state as defined by FPT_PWR_EXT.1**.

*Application Note: The TOE may end up in a non-Compliant power saving state indistinguishable from a Compliant power state (e.g. as result of sudden and/or unexpected power loss). For those scenarios, the TOE or the Operational Environment guidance documentation must provide procedure(s) to support destruction of key material (e.g. automated reboot with memory clearing in early stages of the system's power-on sequence).*

*FCS_KYC_EXT.1 Key Chaining (Initiator)*

**FCS_KYC_EXT.1.1** The TSF shall maintain a key chain of: [selection:

- one, using a submask as the BEV;
- intermediate keys originating from one or more submask(s) to the BEV using the following method(s): [selection:
  - o key derivation as specified in FCS_KDF_EXT.1,
  - o key wrapping as specified in FCS_COP.1(d),
  - o key combining as specified in FCS_SMC_EXT.1,
  - o key transport as specified in FCS_COP.1(e),
  - o key encryption as specified in FCS_COP.1(g)]]

while maintaining an effective strength of [selection: 128 bits, 256 bits] for symmetric keys and an effective strength of [selection: not applicable, 112 bits, 128 bits, 192 bits, 256 bits] for asymmetric keys.

**FCS_KYC_EXT.1.2** The TSF shall provide at least a [selection: 128 bit, 256 bit] BEV to [*assignment: one or more external entities*] [selection:

- after the TSF has successfully performed the validation process as specified in FCS_VAL_EXT.1,
- without validation taking place].

*Application Note: Key Chaining is the method of using multiple layers of encryption keys to ultimately secure the BEV. The number of intermediate keys will vary – from one (e.g., taking*

*the conditioned password authorization factor and directly using it as the BEV) to many. This applies to all keys that contribute to the ultimate wrapping or derivation of the BEV; including those in areas of protected storage (e.g. TPM stored keys, comparison values).*

*Multiple key chains to the BEV are allowed, as long as all chains meet the key chain requirement.*

*The BEV is considered to be equivalent to keying material and therefore additional checksums or similar values are not the BEV, even if they are sent with the BEV.*

*Once the ST author has selected a method to create the chain (either by deriving keys or unwrapping them or encrypting keys or using RSA Key Transport), they pull the appropriate requirement out of Appendix B. It is allowable for an implementation to use any or all methods.*

*For FCS_KYC_EXT.1.2, the validation process is defined in FCS_VAL_EXT.1, Appendix B. If that selection is made by the ST author, then FCS_VAL_EXT.1 is included in the body of the ST.*

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

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

**FCS_SNI_EXT.1.1** The TSF shall [selection: use no salts, use salts that are generated by a [selection: DRBG as specified in FCS_RBG_EXT.1, DRBG provided by the host platform]].

**FCS_SNI_EXT.1.2** The TSF shall use [selection: no nonces, unique nonces with a minimum size of [*64*] bits].

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

- CBC: IVs shall be non-repeating and unpredictable;
- CCM: Nonce shall be non-repeating and unpredictable;
- XTS: No IV. Tweak values shall be non-negative integers, assigned consecutively, and starting at an arbitrary non-negative integer;
- 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 only have to be unique. FCS_SNI_EXT.1.3 specifies how the IV should be handled for each encryption mode. CBC, XTS, and GCM are allowed for AES encryption of the data. AES-CCM is an allowed mode for Key Wrapping.*

## 5.4    Class: Security Management (FMT)

*FMT_MOF.1 Management of Functions Behavior*

**FMT_MOF.1.1** The TSF shall restrict the ability to [<u>modify the behaviour of</u>] the functions [*use of Compliant power saving state*] to [*authorized administrators*].

*Application Note: "Modify the behaviour of" refers to any change in how or when a Compliant power state may occur. Only privileged users are allowed to enable or disable Compliant power saving state(s) via modification of "use of Compliant power saving state" function.*

*FMT_SMF.1 Specification of Management Functions*

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

  *a) forwarding requests to change the DEK to the EE,*
  *b) forwarding requests to cryptographically erase the DEK to the EE,*
  *c) allowing authorized users to change authorization factors or set of authorization*
    *factors used,*
  *d) initiate TOE firmware/software updates,*
  **e)** <u>*[selection: no other functions, specify the power saving state properties, define the*</u>
    <u>*allowable power saving states, generate authorization factors using the TSF RBG,*</u>
    <u>*configure authorization factors, configure cryptographic functionality, disable key*</u>
    <u>*recovery functionality, securely update the public key needed for trusted update,*</u>
    <u>*configure the number of failed validation attempts required to trigger corrective*</u>
    <u>*behavior, configure the corrective behavior to issue in the event of an excessive*</u>
    <u>*number of failed validation attempts, [assignment: other management functions*</u>
    <u>*provided by the TSF]]*</u>].

*Application Note: The intent of this requirement is to express the management capabilities that the TOE possesses. This means that the TOE must be able to perform the listed functions. Item (e) is used to specify functionality that may be included in the TOE, but is not required to conform to the cPP. "Configure cryptographic functionality" could include key management functions; for example, the BEV will be wrapped or encrypted, and the EE will need to unwrap or decrypt the BEV.  In item e, if no other management functions are provided (or claimed), then "no other functions" should be selected.*

*Changing the DEK would require the data to be re-encrypted with the new DEK, but allows the user the ability to generate new DEKs.*

*For the purposes of this document, key sanitization means to destroy the DEK, using one of the approved destruction methods. In some implementations, changing the DEK could be the same functionality as cryptographically erasing the DEK.*

## 5.5 Class: Protection of the TSF (FPT)

*FPT_KYP_EXT.1 Protection of Key and Key Material*

**FPT_KYP_EXT.1.1** The TSF shall [selection: not store keys in non-volatile memory, only store keys in non-volatile memory when wrapped, as specified in FCS_COP.1(d), or encrypted, as specified in FCS_COP.1(g) or FCS_COP.1(e)], unless the key meets any one of following criteria [selection:

- The plaintext key is not part of the key chain as specified in FCS_KYC_EXT.1.
- The plaintext key will no longer provide access to the encrypted data after initial provisioning.
- The plaintext key is a key split that is combined as specified in FCS_SMC_EXT.1, and the other half of the key split is [selection:
    - wrapped as specified in FCS_COP.1(d),
    - encrypted as specified in FCS_COP.1(g) or FCS_COP.1(e),
    - derived and not stored in non-volatile memory].
- The plaintext key is stored on an external storage device for use as an authorization factor.
- The plaintext key is [selection:
    - used to wrap a key as specified in FCS_COP.1(d),
    - encrypted as specified in FCS_COP.1(g) or FCS_COP.1(e)] that is already [selection:
        - wrapped as specified in FCS_COP.1(d),
        - encrypted as specified in FCS_COP.1(g) or FCS_COP.1(e)]].

*Application Note: The plaintext key storage in non-volatile memory is allowed for several reasons. If the keys exist within protected memory that is not user accessible on the TOE or OE, the only methods that allow it to play a security relevant role for protecting the BEV or the DEK are if it is a key split or providing additional layers of wrapping or encryption on keys that have already been protected.*

*FPT_PWR_EXT.1 Power Saving States*

**FPT_PWR_EXT.1.1** The TSF shall define the following Compliant power saving states: [selection: choose at least one of: S3, S4, G2(S5), G3, [*assignment: other power saving states*]].

*Application Note: Power saving states S3, S4, G2(S5), G3 are defined by the Advanced Configuration and Power Interface (ACPI) standard.*

*FPT_PWR_EXT.2 Timing of Power Saving States*

**FPT_PWR_EXT.2.1** For each Compliant power saving state defined in FPT_PWR_EXT.1.1, the TSF shall enter the Compliant power saving state when the following conditions occur: [user-initiated request], [selection: choose at least one of: system

1 shutdown, user inactivity, request initiated by remote management system, [*assignment:*
2 *other conditions*], no other conditions].

3 *Application Note: If volatile memory is not cleared as part of an unexpected power shutdown*
4 *sequence then guidance documentation must define mitigation activities (e.g. how long users*
5 *should wait after an unexpected power-down before volatile memory can be considered*
6 *cleared).*

7 *FPT_TUD_EXT.1 Trusted Update*

8 **FPT_TUD_EXT.1.1 Refinement:** The TSF shall provide [*authorized users*] the ability to
9 query the current version of the TOE **[selection: software, firmware]** ~~software/firmware~~.

10 **FPT_TUD_EXT.1.2 Refinement:** The TSF shall provide [*authorized users*] the ability to
11 initiate updates to TOE **[selection: software, firmware]** ~~software/firmware~~.

12 **FPT_TUD_EXT.1.3 Refinement:** The TSF shall verify updates to the TOE software using a
13 [digital signature **as specified in FCS_COP.1(a)**] by the manufacturer prior to installing
14 those updates.

15 *Application Note: While this component requires the TOE to implement the update*
16 *functionality itself, it is acceptable to perform the cryptographic checks using functionality*
17 *available in the Operational Environment.*

# 1  6. Security Assurance Requirements

2  This cPP identifies the Security Assurance Requirements (SARs) to frame the extent to which
3  the evaluator assesses the documentation applicable for the evaluation and performs
4  independent testing. Individual Evaluation Activities to be performed are specified in
5  *Supporting Document (Mandatory Technical Document) Full Drive Encryption:*
6  *Authorization Acquisition September 2016*.

7  **Note to ST authors**: **There is a selection in the ASE_TSS that must be completed. One**
8  **cannot simply reference the SARs in this cPP.**

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

18
*Table 3: TOE Security Assurance Requirements*

| Functional Class | Functional 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) |

19

## 20  6.1  ASE: Security Target

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

24  The SFRs in this cPP allow for conformant implementations to incorporate a wide range of
25  acceptable key management approaches as long as basic principles are satisfied. Given the
26  criticality of the key management scheme, this cPP requires the developer to provide a

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

5   In addition, if the TOE includes a random bit generator Appendix D provides a description of
6   the information expected to be provided regarding the quality of the entropy.

7   **ASE_TSS.1.1C Refinement:** The TOE summary specification shall describe how the TOE
8   meets each SFR**, including a proprietary Key Management Description (Appendix E),
9   and [selection: Entropy Essay, list of all of 3rd party software libraries (including
10  version numbers), 3rd party hardware components (including model/version numbers),
11  no other cPP specified proprietary documentation].**

## 12   6.2    ADV: Development

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

### 16   6.2.1   Basic Functional Specification (ADV_FSP.1)

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

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

## 29   6.3    AGD: Guidance Documentation

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

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

36      •      instructions to successfully install the TSF in that environment; and

1        •       instructions to manage the security of the TSF as a product and as a component of
2                the larger operational environment; and

3        •       Instructions to provide a protected administrative capability.

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

### 6.3.1  Operational User Guidance (AGD_OPE.1)

7  The operational user guidance does not have to be contained in a single document. Guidance
8  to users, administrators and application developers can be spread among documents or web
9  pages.

10  The developer should review the Evaluation Activities contained in the SD to ascertain the
11  specifics of the guidance that the evaluator will be checking for. This will provide the
12  necessary information for the preparation of acceptable guidance.

### 6.3.2  Preparative Procedures (AGD_PRE.1)

14  As with the operational guidance, the developer should look to the Evaluation Activities to
15  determine the required content with respect to preparative procedures.

## 6.4    Class ALC: Life-cycle Support

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

### 6.4.1  Labelling of the TOE (ALC_CMC.1)

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

### 6.4.2  TOE CM Coverage (ALC_CMS.1)

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

## 6.5    Class ATE: Tests

32  Testing is specified for functional aspects of the system as well as aspects that take advantage
33  of design or implementation weaknesses. The former is done through the ATE_IND family,

1 while the latter is through the AVA_VAN family. For this cPP, testing is based on advertised
2 functionality and interfaces with dependency on the availability of design information. One
3 of the primary outputs of the evaluation process is the test report as specified in the following
4 requirements.

### 5 6.5.1 Independent Testing – Conformance (ATE_IND.1)

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

## 13 6.6 Class AVA: Vulnerability Assessment

14 For the current generation of this cPP, the iTC is expected to survey open sources to discover
15 what vulnerabilities have been discovered in these types of products and provide that content
16 into the AVA_VAN discussion. In most cases, these vulnerabilities will require
17 sophistication beyond that of a basic attacker. This information will be used in the
18 development of future Protection Profiles.

### 19 6.6.1 Vulnerability Survey (AVA_VAN.1)

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

# 1 Appendix A: Optional Requirements

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

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

10 Some of the requirements in this section are iterated, but since the ST author is responsible
11 for incorporating the appropriate requirements from the appendices into the body of their ST,
12 the correct iteration numbering is left to the ST author.

## 13 A.1 Internal Cryptographic Implementation

14 As indicated in the body of this cPP, it is acceptable for the TOE to either directly implement
15 cryptographic functionality that supports the drive encryption/decryption process, or to use
16 that functionality in the Operational Environment (for example, calling an Operating System's
17 cryptographic provider interface; a third-party cryptographic library; or a hardware
18 cryptographic accelerator). However, each one of these SFRs that can optionally be
19 implemented by the Operational Environment are also considered to be 'selection-based'
20 SFRs due to the fact that their functionality is contingent on the ST author make certain
21 selections in other SFRs. Because of this, these SFRs have been placed in Appendix B. Note
22 however that there is still an expectation that some of these functions may be provided by the
23 Operational Environment, in which case it is acceptable to omit the SFRs in question so long
24 as the ST author can provide evidence that the Operational Environment will include a
25 cryptographic interface to the TSF that allows for secure usage of these functions, and that
26 the functions have been validated to the same level of rigor as is described in [SD].

27 If all of the cryptographic functionality is implemented by the TSF and the TOE does not rely
28 on its Operational Environment to provide any cryptographic services, the ST author shall
29 omit OE.STRONG_ENVIRONMENT_CRYPTO and its corresponding assumption since the
30 environment does not need to satisfy the objective in this case.

## 31 A.2 TSF Self-Testing

32 In order to ensure that any cryptographic primitives provided by the TOE is functioning
33 properly, it is necessary for the TSF to provide a self-test function that is used to verify their
34 integrity. The ST author shall include the following SFR if the TSF includes
35 FCS_RBG_EXT.1 or any iteration of FCS_COP.1:

1    *FPT_TST_EXT.1 TSF Testing*

2    **FPT_TST_EXT.1.1** The TSF shall run a suite of the following self-tests [selection: during
3    initial start-up (on power on), at the conditions [*before the function is first invoked*]] to
4    demonstrate the correct operation of the TSF: [*assignment: list of self-tests run by the TSF*].

5    ***Application Note:*** *The tests regarding cryptographic functions implemented in the TOE can*
6    *be deferred, as long as the tests are performed before the function is invoked.*

7    *If FCS_RBG_EXT.1 is implemented by the TOE and according to NIST SP 800-90, the*
8    *evaluator shall verify that the TSS describes health tests that are consistent with section 11.3*
9    *of NIST SP 800-90.*

10    *If any FCS_COP functions are implemented by the TOE, the TSS shall describe the known*
11    *answer self-tests for those functions.*

12    *The evaluator shall verify that the TSS describes, for some set of non-cryptographic functions*
13    *affecting the correct operation of the TSF, the method by which those functions are tested.*
14    *The TSS will describe, for each of these functions, the method by which correct operation of*
15    *the function/component is verified. The evaluator shall determine that all of the identified*
16    *functions/components are adequately tested on start-up.*

# Appendix B: Selection-Based Requirements

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

Note that many of these selection-based SFRs could also be implemented by cryptographic services in the TOE's Operational Environment. If this is the case, it is not necessary to include the SFRs in question so long as the Operational Environment can be shown to provide equivalent functionality.

## B.1 Class: Cryptographic Support (FCS)

If FCS_VAL_EXT.1 is included in the ST, the evaluator shall add the following threat to the ST:

(T.AUTHORIZATION_GUESSING) Threat agents may exercise host software to repeated guess authorization factors, such as passwords and pins. Successful guessing of the authorization factors may cause the TOE to release DEKs or otherwise put it in a state in which it discloses protected data to unauthorized users.

[FCS_VAL_EXT.1]

Rationale: Only valid BEV's [FCS_VAL_EXT.1] are forwarded to the EE [FCS_VAL_EXT.1]. The response to failed validation attempt [FCS_VAL_EXT.1] mitigates the threat of successful authorization factor guessing.

*FCS_CKM.1(a) Cryptographic Key Generation (Asymmetric Keys)*

**FCS_CKM.1.1(a) Refinement:** The TSF shall generate **asymmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm: [*selection:*

- *RSA schemes using cryptographic key sizes of [selection: 2048-bit, 3072-bit, 4096-bit] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3;*

- *ECC schemes using "NIST curves" of [selection: P-256, P-384, P-521] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4;*

- *FFC schemes using cryptographic key sizes of [selection: 2048-bit, 3072-bit, 4096-bit] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.1*

1     ]~~and specified cryptographic key sizes [*assignment: cryptographic key sizes*] that meet~~
2     ~~the following: [*assignment: list of standards*].~~

3     *Application Note: Asymmetric keys may be used to "wrap" a key or submask. This SFR*
4     *should be included by the ST author when making the appropriate selection in FCS_COP.*

5     *Asymmetric Keys may also be used for the key chain. Therefore, the ST author should select*
6     *FCS_CKM.1(a), if Asymmetric key generation is used.*

7     *If the TOE acts as a receiver in the RSA key establishment scheme, the TOE does not need to*
8     *implement RSA key generation.*

9     *For all schemes (RSA schemes, ECC schemes, FFC schemes), an RBG is needed to a)*
10    *generate seeds for RSA and to b) generate private keys directly for ECC and FFC. So*
11    *FCS_RBG_EXT.1 is used together with this SFR. A hash algorithm is also required when the*
12    *key pair generation algorithm is selected based on either Appendix B.3.2 or B.3.5 of FIPS*
13    *186-4. So in such case, FCS_COP.1(d) is used together with this SFR.*

14    **FCS_CKM.1(b) Cryptographic Key Generation (Symmetric Keys)**

15    **FCS_CKM.1.1(b) Refinement:** The TSF shall generate **symmetric** cryptographic keys
16    **using a Random Bit Generator as specified in FCS_RBG_EXT.1** and specified
17    cryptographic key sizes [***selection: 128 bit, 256 bit***] that meet the following: [*no standard*].

18    *Application Note: Symmetric keys may be used to generate keys along the key chain.*
19    *Therefore, the ST author should select FCS_CKM.1(b), if Symmetric key generation is used.*

20    *FCS_COP.1(a) Cryptographic Operation (Signature Verification)*

21    **FCS_COP.1.1(a) Refinement:** The TSF shall perform [*cryptographic signature services*
22    *(verification)*] in accordance with a [***selection:***

23       • ***RSA Digital Signature Algorithm with a key size (modulus)of 2048 bits or greater,***

24       • ***Elliptic Curve Digital Signature Algorithm with a key size of 256 bits or greater***

25    ]

26    that meet the following: [***selection:***

27       • ***FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1***
28         ***v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1-v1_5; ISO/IEC***
29         ***9796-2, 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" [selection: P-256, P-384, P-521]; ISO/IEC 14888-3, Section 6.4, for ECDSA schemes***

].

*Application Note: 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.*

*FCS_COP.1(b) Cryptographic Operation (Hash Algorithm)*

**FCS_COP.1.1(b) Refinement:** The TSF shall perform [*cryptographic hashing services*] in accordance with a specified cryptographic algorithm [***selection: SHA-256, SHA-384, SHA-512***] and cryptographic key sizes [*assignment: cryptographic key sizes*] that meet the following: [*ISO/IEC 10118-3:2004*].

*Application Note: The hash selection should be consistent with the overall strength of the algorithm used for FCS_COP.1(a). For example, SHA-256 should be chosen for 2048-bit RSA or ECC with P-256, SHA-384 should be chosen for 3072-bit RSA, 4096-bit RSA, or ECC with P-384, and SHA-512 should be chosen for ECC with P-521. The selection of the standard is made based on the algorithms selected.*

*FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm)*

**FCS_COP.1.1(c) Refinement:** The TSF shall perform [*keyed-hash message authentication*] in accordance with a specified cryptographic algorithm [***selection: HMAC-SHA-256, HMAC-SHA-384, 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: The key size [k] in the assignment 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.*

*FCS_COP.1(d) Cryptographic Operation (Key Wrapping)*

**FCS_COP.1.1(d) Refinement:** 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: [*AES as specified in ISO/IEC 18033-3, [**selection: NIST SP 800-38F, ISO/IEC 19772, no other standards**]*].

*Application Note: This requirement is 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.1.*

1   *FCS_COP.1(e) Cryptographic Operation (Key Transport)*

2   **FCS_COP.1.1(e) Refinement:** The TSF shall perform [*key transport*] in accordance with a
3   specified cryptographic algorithm [*RSA in the following modes **[selection: KTS-OAEP, KTS-**
4   ***KEM-KWS]***] and the cryptographic key size [***selection: 2048 bits, 3072 bits***] that meet the
5   following: [*NIST SP 800-56B, Revision 1*].

6   *Application Note: This requirement is used in the body of the ST if the ST author chooses to*
7   *use key transport in the key chaining approach that is specified in FCS_KYC_EXT.1.*

8   *FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption)*

9   **FCS_COP.1.1(f) Refinement:** The TSF shall perform [*data encryption and decryption*] in
10  accordance with a specified cryptographic algorithm [*AES used in **[selection: CBC, GCM,***
11  ***XTS]** mode*] and cryptographic key sizes [***selection: 128 bits, 256 bits***] that meet the
12  following: [*AES as specified in ISO /IEC 18033-3, **[selection: CBC as specified in ISO/IEC***
13  ***10116, GCM as specified in ISO/IEC 19772, XTS as specified in IEEE 1619]***].

14  *Application Note: The intent of this requirement in the context of this cPP is to provide a*
15  *SFR that expresses the appropriate symmetric encryption/decryption algorithms suitable for*
16  *use in the TOE. If the ST author incorporates the validation requirement (FCS_VAL_EXT.1)*
17  *and chooses to select the option to decrypt a known value and perform a comparison, this is*
18  *the requirement used to specify the algorithm, modes, and key sizes the ST author can choose*
19  *from. Or, this requirement is used in the body of the ST if the ST author chooses to use AES*
20  *encryption/decryption for protecting the keys as part of the key chaining approach that is*
21  *specified in FCS_KYC_EXT.1.*

22  *When the XTS mode is selected, a cryptographic key of 256-bit or of 512-bit is allowed as*
23  *specified in IEEE 1619. XTS-AES key is divided into two AES keys of equal size - for*
24  *example, AES-128 is used as the underlying algorithm, when 256-bit key and XTS mode are*
25  *selected. AES-256 is used when a 512-bit key and XTS mode are selected.*

26  *FCS_COP.1(g) Cryptographic Operation (Key Encryption)*

27  **FCS_COP.1.1(g) Refinement:** The TSF shall perform [*key encryption and decryption*] in
28  accordance with a specified cryptographic algorithm [*AES used in **[selection: CBC, GCM]***
29  *mode*] and cryptographic key sizes [***selection: 128 bits, 256 bits***] that meet the following:
30  [*AES as specified in ISO /IEC 18033-3, **[selection: CBC as specified in ISO/IEC 10116,***
31  ***GCM as specified in ISO/IEC 19772]***].

32  *Application Note: This requirement is used in the body of the ST if the ST author chooses to*
33  *use AES encryption/decryption for protecting the keys as part of the key chaining approach*
34  *that is specified in FCS_KYC_EXT.1.*

1   *FCS_KDF_EXT.1 Cryptographic Key Derivation*

2   **FCS_KDF_EXT.1.1** The TSF shall accept [selection: a RNG generated submask as specified
3   in FCS_RBG_EXT.1, a conditioned password submask, imported submask] to derive an
4   intermediate key, as defined in [selection:
5      • NIST SP 800-108 [selection: KDF in Counter Mode, KDF in Feedback Mode, KDF
6         in Double-Pipeline Iteration Mode],
7      • NIST SP 800-132],
8   using the keyed-hash functions specified in FCS_COP.1(c), such that the output is at least of
9   equivalent security strength (in number of bits) to the BEV.

10  *Application Note: This requirement is used in the body of the ST if the ST author chooses to*
11  *use key derivation in the key chaining approach that is specified in FCS_KYC_EXT.1.*

12  *This requirement establishes acceptable methods for generating a new random key or an*
13  *existing submask to create a new key along the key chain.*

14  *FCS_PCC_EXT.1 Cryptographic Password Construct and Conditioning*

15  **FCS_PCC_EXT.1.1** A password used by the TSF to generate a password authorization factor
16  shall enable up to [*assignment: positive integer of 64 or more*] characters in the set of {upper case
17  characters, lower case characters, numbers, and [*assignment: other supported special*
18  *characters*]} and shall perform Password-based Key Derivation Functions in accordance with a
19  specified cryptographic algorithm HMAC-[selection: SHA-256, SHA-512], with [*assignment:*
20  *positive integer of 1000 or more*] iterations, and output cryptographic key sizes [selection: 128
21  bits, 256 bits] that meet the following: [*NIST SP 800-132*].

22  *Application Note: The password is represented on the host machine as a sequence of*
23  *characters whose encoding depends on the TOE and the underlying OS. This sequence must*
24  *be conditioned into a string of bits that forms the submask to be used as input into the key*
25  *chain. Conditioning can be performed using one of the identified hash functions or the*
26  *process described in NIST SP 800-132; the method used is selected by the ST author. If 800-*
27  *132 conditioning is specified, then the ST author fills in the number of iterations that are*
28  *performed. 800-132 also requires the use of a pseudo-random function (PRF) consisting of*
29  *HMAC with an approved hash function. The ST author selects the hash function used which*
30  *also includes the appropriate requirements for HMAC.*

31  *FCS_RBG_EXT.1 Cryptographic Operation (Random Bit Generation)*

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

35  **FCS_RBG_EXT.1.2** The deterministic RBG shall be seeded by at least one entropy source
36  that accumulates entropy from [selection:
37     • [*assignment: number of software-based sources*] software-based noise source(s),
38     • [*assignment: number of hardware-based sources*] hardware-based noise source(s)]

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-256, 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_DRBG 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 to which the TSF is compliant.*

*In the first selection in FCS_RBG_EXT.1.2 the ST author fills in how many entropy sources are used for each type of entropy 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 DRBG and if the DRBG is included in the TOE, the developer is required to provide the entropy description outlined in Appendix D. The documentation \*and tests\* required in the Evaluation Activity for this element necessarily cover each source indicated in FCS_RBG_EXT.1.2. Individual contributions to the entropy pool may be combined to provide the minimum amount of entropy as long as the Entropy Documentation demonstrates that entropy from each of these individual sources is generated independently.*

*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-384, SHA-512] to generate an [*intermediary key or BEV*].

*Application Note: 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(b).*

*FCS_VAL_EXT.1 Validation*

**FCS_VAL_EXT.1.1** The TSF shall perform validation of the [selection: submask, intermediate key, BEV] using the following method(s): [selection:

- key wrap as specified in FCS_COP.1(d);

- hash the [selection: submask, intermediate key, BEV] as specified in [selection: FCS_COP.1(b), FCS_COP.1(c)] and compare it to a stored hashed [selection: submask, intermediate key, BEV];
- decrypt a known value using the [selection: submask, intermediate key, BEV] as specified in FCS_COP.1(f) and compare it against a stored known value]

**FCS_VAL_EXT.1.2** The TSF shall require validation of the [BEV] prior to [*forwarding the BEV to the EE*].

**FCS_VAL_EXT.1.3** The TSF shall [selection:

- [*perform a key sanitization of the DEK*] upon a configurable number of consecutive failed validation attempts,
- institute a delay such that only [*assignment: ST author specified number of attempts*] can be made within a 24 hour period,
- block validation after [*assignment: ST author specified number of attempts*] of consecutive failed validation attempts,
- require power cycle/reset the TOE after [*assignment: ST author specified number of attempts*] of consecutive failed validation attempts].

*Application Note: The purpose of performing secure validation is to not expose any material that might compromise the submask(s). For the selections in FCS_VAL_EXT.1.1, the ST author must clarify in the KMD which specific entities are referred to in this SFR if multiple entities of a type exist.*

*The TOE validates the submask(s) (e.g., authorization factor(s)) prior to presenting the BEV to the EE. When a password is used as an authorization factor, it is conditioned before any attempts to validate. In cases where validation of the authorization factor(s) fails, the product will not forward a BEV to EE.*

*When the key wrap in FCS_COP.1(d) is used, the validation is performed inherently.*

*The delay must be enforced by the TOE, but this requirement is not intended to address attacks that bypass the product (e.g. attacker obtains hash value or "known" crypto value and mounts attacks outside of the TOE, such as a third party password crackers). The cryptographic functions (i.e., hash, decryption) performed are those specified in FCS_COP.1(b), FCS_COP.1(c), and FCS_COP.1(f).*

*The ST author may need to iterate this requirement if multiple authentication factors are used, and either different methods are used to validate, or in some cases one or more authentication factors may be validated, and one or more are not validated.*

# 1 Appendix C: Extended Component Definitions

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

4 Note that several of the extended requirements used for this cPP have dependencies on SFRs
5 that are iterated in the cPP (e.g. FCS_COP.1(d)). The reader is advised that the SFR names
6 for these dependencies may differ if the same extended components are used in other
7 Protection Profiles.

## 8 C.1 Background and Scope

9 This document provides a definition for all of the extended components used in this cPP.
10 These components are identified in the following table:

11                                    *Table 4: Extended Components*

| Functional Class | Functional Components |
|---|---|
| Cryptographic Support (FCS) | FCS_AFA_EXT Authorization Factor Acquisition |
| | FCS_CKM_EXT Cryptographic Key Management |
| | FCS_KDF_EXT Cryptographic Key Derivation |
| | FCS_KYC_EXT Key Chaining |
| | FCS_PCC_EXT Cryptographic Password Construction and Conditioning |
| | FCS_RBG_EXT Cryptographic Operation (Random Bit Generation) |
| | FCS_SMC_EXT Submask Combining |
| | FCS_SNI_EXT Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) |
| | FCS_VAL_EXT Validation of Cryptographic Elements |
| Protection of the TSF (FPT) | FPT_KYP_EXT Key and Key Material Protection |
| | FPT_TST_EXT TSF Testing |
| | FPT_TUD_EXT Trusted Update |

12 Note that several of the extended components define dependencies on iterated Part 2 SFRs
13 that are defined in this cPP. This definition mandates that these dependencies be included in a
14 PP that claims the SFR but it does not mandate that the dependent SFRs are defined using the
15 same iteration identifiers (e.g. inclusion of FCS_KDF_EXT.1 does not require the dependent
16 SFR for keyed-hash message authentication to be identified specifically as FCS_COP.1(c),
17 only that an FCS_COP.1 iteration exists and defines the same behavior as what this cPP
18 defines as FCS_COP.1(c)).

## 19 C.2 Extended Component Definitions

20 *FCS_AFA_EXT Authorization Factor Acquisition*

21 **Family Behavior**

22 Components in this family address the ability for the TOE to accept a variety of authorization
23 factors.

1    **Component Leveling**



2

3    FCS_AFA_EXT.1, Authorization Factor Acquisition, requires authorization factors to be
4    accepted by the TOE.

5    FCS_AFA_EXT.2, Timing of Authorization Factor Acquisition, defines situations in which
6    the TOE is to accept authorization factors.

7    **Management: FCS_AFA_EXT.1**

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

9       • Change the authorization factors to be used
10      • Generate external authorization factors using the TSF DRBG

11   **Audit: FCS_AFA_EXT.1**

12   There are no auditable events foreseen.

13   **Management: FCS_AFA_EXT.2**

14   There are no management activities foreseen.

15   **Audit: FCS_AFA_EXT.2**

16   There are no auditable events foreseen.

17   **FCS_AFA_EXT.1 Authorization Factor Acquisition**

18   Hierarchical to:        No other components

19   Dependencies:        No dependencies

20   **FCS_AFA_EXT.1.1** The TSF shall accept the following authorization factors: [selection:

21      • a submask derived from a password authorization factor conditioned as defined in
22        FCS_PCC_EXT.1,

23      • an external Smartcard factor that is at least the same bit-length as the DEK, and is
24        protecting a submask that is [selection: generated by the TOE (using the RBG as

1        specified in FCS_RBG_EXT.1), generated by the Host Platform] protected using
2        RSA with key size [selection: 2048 bits, 3072 bits, 4096 bits], with user presence
3        proved by presentation of the smartcard and [selection: none, an OE defined PIN,
4        a configurable PIN],

5        •   an external USB token factor that is at least the same security strength as the BEV,
6          and is providing a submask generated by the TOE, using the RBG as specified in
7          FCS_RBG_EXT.1,

8        •   an external USB token factor that is at least the same security strength as the BEV,
9          and is providing a submask generated by the Host Platform

10        ].

11 **FCS_AFA_EXT.2 Authorization Factor Acquisition**

12 Hierarchical to:      No other components

13 Dependencies:      FCS_AFA_EXT.1 Authorization Factor Acquisition,
14                          FPT_PWR_EXT.1 Power Saving States

15 **FCS_AFA_EXT.2.1** The TSF shall reacquire the authorization factor(s) specified in
16 FCS_AFA_EXT.1 upon transition from any Compliant power saving state specified in
17 FPT_PWR_EXT.1 prior to permitting access to plaintext data.

18 *FCS_CKM_EXT Cryptographic Key Management*

19 **Family Behavior**

20 Cryptographic keys must be managed throughout their life cycle. This family is intended to
21 support that lifecycle and consequently defines requirements for the following activities:
22 cryptographic key generation, cryptographic key distribution, cryptographic key access and
23 cryptographic key destruction. This family should be included whenever there are functional
24 requirements for the management of cryptographic keys.

25 The creation of this family is necessary because CC Part 2 provides the ability to specify the
26 method of key destruction but does not define SFRs for the timing of key destruction or the
27 ability to implement multiple key destruction methods.

28 **Component Leveling**



29
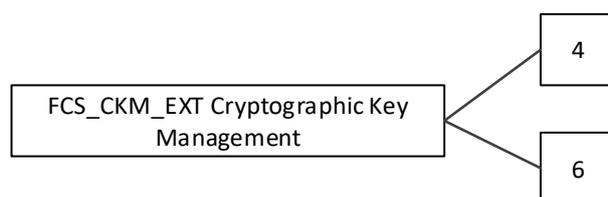
1 FCS_CKM_EXT.4, Key and Key Material Destruction, requires the TSF to specify
2 circumstances when keys are destroyed (as opposed to the actual method of destruction,
3 which is defined in CC Part 2 as FCS_CKM.4). The number 4 was chosen to reflect the
4 similarity between the two SFRs.

5 FCS_CKM_EXT.6, Cryptographic Key Destruction Types, provides the TOE with the ability
6 to select between multiple methods of key destruction.

7 **Management: FCS_CKM_EXT.4**

8 No specific management functions are identified.

9 **Audit: FCS_CKM_EXT.4**

10 There are no auditable events foreseen.

11 **Management: FCS_CKM_EXT.4**

12 No specific management functions are identified.

13 **Audit: FCS_CKM_EXT.4**

14 There are no auditable events foreseen.

15 **FCS_CKM_EXT.4 Cryptographic Key and Key Material Destruction**

16 Hierarchical to:      No other components

17 Dependencies:      No dependencies

18 **FCS_CKM_EXT.4.1** The TSF shall destroy all keys and key material when no longer
19 needed.

20 **FCS_CKM_EXT.6 Cryptographic Key Destruction Types**

21 Hierarchical to:      No other components

22 Dependencies:      FCS_CKM.4 Cryptographic Key Destruction

23 **FCS_CKM_EXT.6.1** The TSF shall use [*assignment: two or more iterations of FCS_CKM.4*
24 *defined elsewhere in the Security Target*] key destruction methods.

25 *FCS_KDF_EXT Cryptographic Key Derivation*

26 **Family Behavior**

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

**Component Leveling**

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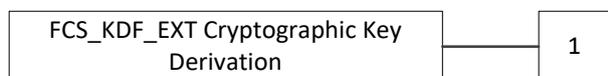
5 FCS_KDF_EXT.1, Cryptographic Key Derivation, requires the TSF to derive intermediate
6 keys from submasks using the specified hash functions.

**Management: FCS_KDF_EXT.1**

No specific management functions are identified.

**Audit: FCS_KDF_EXT.1**

There are no auditable events foreseen.

*FCS_KDF_EXT.1 Cryptographic Key Derivation*

Hierarchical to:        No other components

Dependencies:        FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm)

**FCS_KDF_EXT.1.1** The TSF shall accept [selection: a RNG generated submask as specified
in FCS_RBG_EXT.1, a conditioned password submask, imported submask] 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(c), such that the output is at least of
equivalent security strength (in number of bits) to the BEV.

*FCS_KYC_EXT Key Chaining*

**Family Behavior**

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

**Component Leveling**
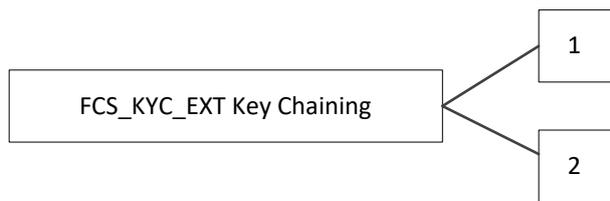
1

2 FCS_KYC_EXT.1, Key Chaining (Initiator), requires the TSF to maintain a key chain for a
3 BEV that is provided to a component external to the TOE.

4 FCS_KYC_EXT.2, Key Chaining (Recipient), requires the TSF to be able to accept a BEV
5 that is then chained to a DEK used by the TSF through some method.

6 Note that this cPP does not include FCS_KYC_EXT.2; it is only included here to provide a
7 complete definition of the FCS_KYC_EXT family.

8 **Management: FCS_KYC_EXT.1**

9 No specific management functions are identified.

10 **Audit: FCS_KYC_EXT.1**

11 There are no auditable events foreseen.

12 **Management: FCS_KYC_EXT.2**

13 No specific management functions are identified.

14 **Audit: FCS_KYC_EXT.2**

15 There are no auditable events foreseen.

16 **FCS_KYC_EXT.1 Key Chaining (Initiator)**

17 Hierarchical to:     No other components

18 Dependencies:     FCS_CKM.1(a) Cryptographic Key Generation (Asymmetric Keys),
19                   FCS_CKM.1(b) Cryptographic Operation (Symmetric Keys),
20                   FCS_COP.1(d) Cryptographic Operation (Key Wrapping),
21                   FCS_COP.1(e) Cryptographic Operation (Key Transport),
22                   FCS_COP.1(g) Cryptographic Operation (Key Encryption),
23                   FCS_SMC_EXT.1 Submask Combining,
24                   FCS_VAL_EXT.1 Validation
25                   FCS_VAL_EXT.2 User Validation

26 **FCS_KYC_EXT.1.1** The TSF shall maintain a key chain of: [selection:

27     • one, using a submask as the BEV;

- intermediate keys generated by the TSF using the following method(s): [selection:
  - asymmetric key generation as specified in FCS_CKM.1(a),
  - symmetric key generation as specified in FCS_CKM.1(b)];
- intermediate keys originating from one or more submask(s) to the BEV using the following method(s): [selection:
  - key derivation as specified in FCS_KDF_EXT.1,
  - key wrapping as specified in FCS_COP.1(d),
  - key combining as specified in FCS_SMC_EXT.1,
  - key transport as specified in FCS_COP.1(e),
  - key encryption as specified in FCS_COP.1(g)]]

while maintaining an effective strength of [selection: 128 bits, 256 bits] for symmetric keys and an effective strength of [selection: not applicable, 112 bits, 128 bits, 192 bits, 256 bits] for asymmetric keys.

**FCS_KYC_EXT.1.2** The TSF shall provide a at least [selection: 128 bit, 256 bit] BEV to [*assignment: one or more external entities*] [selection:
- after the TSF has successfully performed the validation process as specified in FCS_VAL_EXT.1,
- without validation taking place].

*Application Note: Key Chaining is the method of using multiple layers of encryption keys to ultimately secure the BEV. The number of intermediate keys will vary – from one (e.g., taking the conditioned password authorization factor and directly using it as the BEV) to many. This applies to all keys that contribute to the ultimate wrapping or derivation of the BEV; including those in areas of protected storage (e.g. TPM stored keys, comparison values).*

**FCS_KYC_EXT.2 Key Chaining (Recipient)**

Hierarchical to:      No other components

Dependencies:        No dependencies

**FCS_KYC_EXT.2.1** The TSF shall accept a BEV of at least [selection: 128 bits, 256 bits] from [*assignment: one or more external entities*].

**FCS_KYC_EXT.2.2** The TSF shall maintain a chain of intermediary keys originating from the BEV to the DEK using the following method(s): [selection:

- asymmetric key generation as specified in FCS_CKM.1(a)
- symmetric key generation as specified in FCS_CKM.1(b)
- key derivation as specified in FCS_KDF_EXT.1,
- key wrapping as specified in FCS_COP.1(d),
- key transport as specified in FCS_COP.1(e),
- key encryption as specified in FCS_COP.1(g)]

1 while maintaining an effective strength of [selection: 128 bits, 256 bits] while maintaining an
2 effective strength of [selection: 128 bits, 256 bits] for symmetric keys and an effective
3 strength of [selection: not applicable, 112 bits, 128 bits, 192 bits, 256 bits] for asymmetric
4 keys.

5 *Application Note: Key Chaining is the method of using multiple layers of encryption keys to*
6 *ultimately secure the protected data encrypted on the drive. The number of intermediate keys*
7 *will vary – from one (e.g., using the BEV as a key encrypting key (KEK)) to many. This*
8 *applies to all keys that contribute to the ultimate wrapping or derivation of the DEK;*
9 *including those in areas of protected storage (e.g. TPM stored keys, comparison values).*

10 *FCS_PCC_EXT Cryptographic Password Construction and Conditioning*

11 **Family Behavior**

12 This family ensures that passwords used to produce the BEV are robust (in terms of their
13 composition) and are conditioned to provide an appropriate-length bit string.

14 **Component Leveling**

15
| FCS_PCC_EXT Cryptographic Password Construction and Conditioning | 1 |
|---|---|
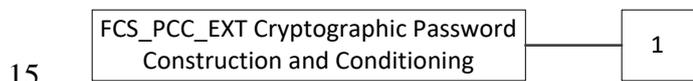
16 FCS_PCC_EXT.1, Cryptographic Password Construction and Conditioning, requires the TSF
17 to accept passwords of a certain composition and condition them appropriately.

18 **Management: FCS_PCC_EXT.1**

19 No specific management functions are identified.

20 **Audit: FCS_PCC_EXT.1**

21 There are no auditable events foreseen.

22 **FCS_PCC_EXT.1 Cryptographic Password Construction and Conditioning**

23 Hierarchical to:        No other components

24 Dependencies:        FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm)

25 **FCS_PCC_EXT.1.1** A password used by the TSF to generate a password authorization factor
26 shall enable up to [*assignment: positive integer of 64 or more*] characters in the set of {upper case
27 characters, lower case characters, numbers, and [*assignment: other supported special*
28 *characters*]} and shall perform Password-based Key Derivation Functions in accordance with a
29 specified cryptographic algorithm HMAC-[selection: SHA-256, SHA-512], with [*assignment:*
30 *positive integer of 1000 or more*] iterations, and output cryptographic key sizes [selection: 128
31 bits, 256 bits] that meet the following: [*assignment: PBKDF recommendation or specification*].

1 *FCS_RBG_EXT Random Bit Generation*

2 **Family Behavior**

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

5 **Component Leveling**

6
```
┌─────────────────────────────────────┐        ┌───────┐
│ FCS_RBG_EXT Random Bit Generation    │────────│   1   │
└─────────────────────────────────────┘        └───────┘
```

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

9 **Management: FCS_RBG_EXT.1**

10 No specific management functions are identified.

11 **Audit: FCS_RBG_EXT.1**

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

14    • Failure of the randomization process

15 **FCS_RBG_EXT.1 Cryptographic Operation (Random Bit Generation)**

16 Hierarchical to:        No other components

17 Dependencies:          FCS_COP.1(b) Cryptographic Operation (Hash Algorithm),
18                        FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm)

19 **FCS_RBG_EXT.1.1** The TSF shall perform all deterministic random bit generation services
20 in accordance with ISO/IEC 18031:2011 using [selection: Hash_DRBG (any),
21 HMAC_DRBG (any), CTR_DRBG (AES)].

22 **FCS_RBG_EXT.1.2** The deterministic RBG shall be seeded by at least one entropy source
23 that accumulates entropy from [selection:
24    • [*assignment: number of software-based sources*] software-based noise source(s),
25    • [*assignment: number of hardware-based sources*] hardware-based noise source(s)]
26 with a minimum of [selection: 128 bits, 256 bits] of entropy at least equal to the greatest
27 security strength, according to ISO/IEC 18031:2011 Table C.1 "Security Strength Table for
28 Hash Functions", of the keys and hashes that it will generate.

1  *Application Note: ISO/IEC 18031:2011contains three different methods of generating*
2  *random numbers; each of these, in turn, depends on underlying cryptographic primitives*
3  *(hash functions/ciphers). The ST author will select the function used, and include the specific*
4  *underlying cryptographic primitives used in the requirement. While any of the identified hash*
5  *functions (SHA-256, SHA-512) are allowed for Hash_DRBG or HMAC_DRBG, only AES-*
6  *based implementations for CTR_DRBG are allowed.*

7  *FCS_SMC_EXT Submask Combining*

8  **Family Behavior**

9  This family specifies the means by which submasks are combined, if the TOE supports more
10 than one submask being used to derive or protect the BEV.

11 **Component Leveling**

12
```
┌─────────────────────────────────┐        ┌───────┐
│ FCS_SMC_EXT Submask Combining   │────────│   1   │
└─────────────────────────────────┘        └───────┘
```
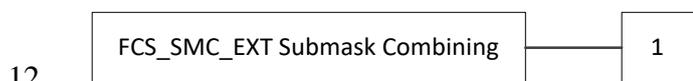
13 FCS_SMC_EXT.1, Submask Combining, requires the TSF to combine the submasks in a
14 predictable fashion.

15 **Management: FCS_SMC_EXT.1**

16 No specific management functions are identified.

17 **Audit: FCS_SMC_EXT.1**

18 There are no auditable events foreseen.

19 **FCS_SMC_EXT.1 Submask Combining**

20 Hierarchical to:        No other components

21 Dependencies:         FCS_COP.1(b) Cryptographic Operation (Hash Algorithm)

22 **FCS_SMC_EXT.1.1** The TSF shall combine submasks using the following method
23 [selection: exclusive OR (XOR), SHA-256, SHA-512] to generate an [*assignment: types of*
24 *keys*].

25 *FCS_SNI_EXT Cryptographic Operation (Salt, Nonce, and Initialization Vector*
26 *Generation*

27 **Family Behavior**

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

1 **Component Leveling**

2

```
┌──────────────────────────────────────┐        ┌─────┐
│ FCS_SNI_EXT Cryptographic Operation (Salt, │────────│  1  │
│ Nonce, and Initialization Vector Generation)│        └─────┘
└──────────────────────────────────────┘
```
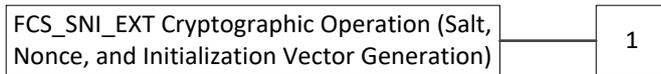
3 FCS_SNI_EXT.1, Cryptographic Operation (Salt, Nonce, and Initialization Vector
4 Generation), requires the generation of salts, nonces, and IVs to be used by the cryptographic
5 components of the TOE to be performed in the specified manner.

6 **Management: FCS_SNI_EXT.1**

7 No specific management functions are identified.

8 **Audit: FCS_SNI_EXT.1**

9 There are no auditable events foreseen.

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

12 Hierarchical to:        No other components

13 Dependencies:        FCS_RBG_EXT.1 Cryptographic Operation (Random Bit Generation)

14 **FCS_SNI_EXT.1.1** The TSF shall [selection: use no salts, use salts that are generated by
15 [selection: DRBG as specified in FCS_RBG_EXT.1, DRBG provided by the host platform]].

16 **FCS_SNI_EXT.1.2** The TSF shall use [selection: no nonces, unique nonces with a minimum
17 size of [*64*] bits].

18 **FCS_SNI_EXT.1.3** The TSF shall create IVs in the following manner [selection:

19 • CBC: IVs shall be non-repeating and unpredictable;
20 • CCM: Nonce shall be non-repeating and unpredictable;
21 • XTS: No IV. Tweak values shall be non-negative integers, assigned consecutively,
22 and starting at an arbitrary non-negative integer;
23 • GCM: IV shall be non-repeating. The number of invocations of GCM shall not exceed
24 $2^{32}$ for a given secret key].

25 FCS_VAL_EXT Validation of Cryptographic Elements

26 **Family Behavior**

27 This family specifies the means by which submasks and/or BEVs are determined to be valid
28 prior to their use.

1 **Component Leveling**
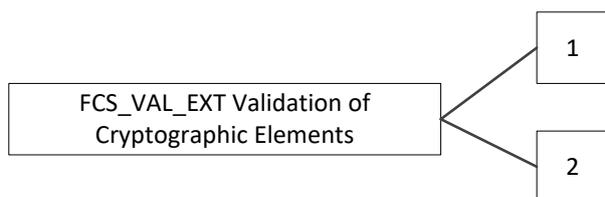
2



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

5 FCS_VAL_EXT.2, User Validation, requires the TSF to validate the legitimacy of a user's
6 request before providing cryptographic data to the user.

7 **Management: FCS_VAL_EXT.1**

8 No specific management functions are identified.

9 **Audit: FCS_VAL_EXT.1**

10 There are no auditable events foreseen.

11 **Management: FCS_VAL_EXT.2**

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

13 • Specification of the validation method used
14 • Configuration of number of failed validation attempts that will be accepted by the
15 TSF
16 • Action taken by the TSF in the event an unacceptable number of failed validation
17 attempts are made

18 **Audit: FCS_VAL_EXT.2**

19 There are no auditable events foreseen.

20 **FCS_VAL_EXT.1 Validation**

21 Hierarchical to:      No other components

22 Dependencies:      FCS_COP.1(b) Cryptographic Operation (Hash Algorithm),
23                          FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm),
24                          FCS_COP.1(d) Cryptographic Operation (Key Wrapping),
25                          FCS_COP.1(f) Cryptographic Operation (AES Data
26                          Encryption/Decryption)

**FCS_VAL_EXT.1.1** The TSF shall perform validation of the [selection: submask, intermediate key, BEV] using the following method(s): [selection:

- key wrap as specified in FCS_COP.1(d);
- hash the [selection: submask, intermediate key, BEV] as specified in [selection: FCS_COP.1(b), FCS_COP.1(c)] and compare it to a stored hashed [selection: submask, intermediate key, BEV];
- decrypt a known value using the [selection: submask, intermediate key, BEV] as specified in FCS_COP.1(f) and compare it against a stored known value]

**FCS_VAL_EXT.1.2** The TSF shall require validation of the [selection: submask, intermediate key, BEV] prior to [*assignment: activity requiring validation*].

**FCS_VAL_EXT.1.3** The TSF shall [selection:

- [*perform a key sanitization of the DEK*] upon a configurable number of consecutive failed validation attempts,
- institute a delay such that only [*assignment: ST author specified number of attempts*] can be made within a 24 hour period,
- block validation after [*assignment: ST author specified number of attempts*] of consecutive failed validation attempts,
- require power cycle/reset the TOE after [*assignment: ST author specified number of attempts*] of consecutive failed validation attempts].

**FCS_VAL_EXT.2 User Validation**

**FCS_VAL_EXT.2.1** The TSF shall perform validation of the [user] by receiving assertion of the user's validity from: [*assignment: Operational Environment component responsible for user authentication*].

**FCS_VAL_EXT.2.2** The TSF shall require validation of the user prior to [*assignment: cryptographic operation or transmission of cryptographic data*].

**FCS_VAL_EXT.2.3** The TSF shall [selection:
- [*assignment: key sanitization activity*] upon receiving a configurable number of consecutive failed validation attempts from the Operational Environment,
- institute a delay such that only [*assignment: ST author specified number of attempts*] can be made within a 24 hour period,
- block validation after [*assignment: ST author specified number of attempts*] of consecutive failed validation attempts,
- require power cycle of or reset the TOE after [*assignment: ST author specified number of attempts*] of consecutive failed validation attempts].

1 *FPT_KYP_EXT Key and Key Material Protection*

2 **Family Behavior**

3 This family requires that key and key material be protected if and when written to non-
4 volatile storage.

5 **Component Leveling**
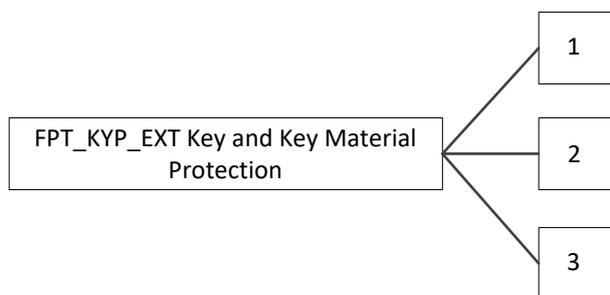


6

7 FPT_KYP_EXT.1, Protection of Key and Key Material, requires the TSF to ensure that no
8 plaintext key or key material are written to non-volatile storage.

9 FPT_KYP_EXT.2, Storage of Protected Key and Key Material, requires the TSF to specify
10 the non-volatile storage location in which encrypted key and key material is stored.

11 FPT_KYP_EXT.3, Attribution of Protected Key and Key Material, requires the TSF to
12 maintain an association between encrypted key and key material and the subjects that are
13 authorized to decrypt and/or use the data.

14 **Management: FPT_KYP_EXT.1**

15 No specific management functions are identified.

16 **Audit: FPT_KYP_EXT.1**

17 There are no auditable events foreseen.

18 **Management: FPT_KYP_EXT.2**

19 No specific management functions are identified.

20 **Audit: FPT_KYP_EXT.2**

21 There are no auditable events foreseen.

22 **Management: FPT_KYP_EXT.3**

23 No specific management functions are identified.

1 **Audit: FPT_KYP_EXT.3**

2 There are no auditable events foreseen.

3 **FPT_KYP_EXT.1 Protection of Key and Key Material**

4 Hierarchical to:      No other components

5 Dependencies:      FCS_COP.1(d) Cryptographic Operation (Key Wrapping),
6                               FCS_COP.1(e) Cryptographic Operation (Key Transport),
7                               FCS_COP.1(g) Cryptographic Operation (Key Encryption),
8                               FCS_KYC_EXT.1 Key Chaining (Initiator),
9                               FCS_KYC_EXT.2 Key Chaining (Recipient),
10                              FCS_SMC_EXT.1 Submask Combining

11 **FPT_ KYP _EXT.1.1** The TSF shall [selection: not store keys in non-volatile memory, only
12 store keys in non-volatile memory when wrapped, as specified in FCS_COP.1(d) or
13 encrypted, as specified in FCS_COP.1(g) or FCS_COP.1(e)], unless the key meets any one of
14 following criteria [selection:

15  • The plaintext key is not part of the key chain as specified in [selection:
16     o FCS_KYC_EXT.1,
17     o FCS_KYC_EXT.2].
18  • The plaintext key will no longer provide access to the encrypted data after initial
19     provisioning.
20  • The plaintext key is a key split that is combined as specified in FCS_SMC_EXT.1,
21     and the other half of the key split is [selection:
22     o wrapped as specified in FCS_COP.1(d),
23     o encrypted as specified in FCS_COP.1(g) or FCS_COP.1(e),
24     o derived and not stored in non-volatile memory].
25  • The plaintext key is stored on an external storage device for use as an authorization
26     factor.
27  • The plaintext key is [selection:
28     o used to wrap a key as specified in FCS_COP.1(d),
29     o encrypted as specified in FCS_COP.1(g) or FCS_COP.1(e)] that is already
30        [selection:
31        ▪ wrapped as specified in FCS_COP.1(d),
32        ▪ encrypted as specified in FCS_COP.1(g) or FCS_COP.1(e)]].

33 **FPT_KYP_EXT.2 Storage of Protected Key and Key Material**

34 Hierarchical to:      No other components

35 Dependencies:      FPT_KYP_EXT.1 Protection of Key and Key Material

**FPT_KYP_EXT.2.1** The TSF shall only store protected key and key material [selection: within the TSF, in a SQL database in the Operational Environment, [*assignment: other key storage location*]].

**FPT_KYP_EXT.3 Attribution of Protected Key and Key Material**

Hierarchical to:        No other components

Dependencies:        FPT_KYP_EXT.1 Protection of Key and Key Material

**FPT_KYP_EXT.3.1** The TSF shall maintain an association between [*assignment: list of key and key material*] and [*assignment: subjects that are authorized to use the identified key and key material*].

*FPT_PWR_EXT Power Management*

**Family Behavior**

This family defines secure behavior of the TSF when the TOE supports multiple power saving states. The use of Compliant power saving states (i.e. power saving states that purge security-relevant data upon entry) is essential for ensuring that state transitions cannot be used as attack vectors to bypass TOE self-protection mechanisms.

**Component Leveling**
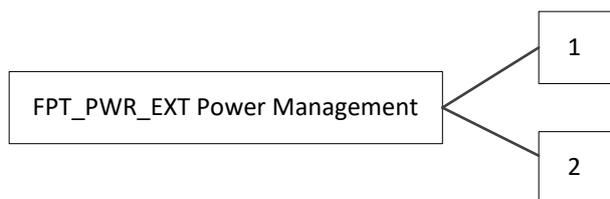


FPT_PWR_EXT.1, Power Saving States, defines the Compliant power saving states that are implemented by the TSF.

FPT_PWR_EXT.2, Timing of Power Saving States, describes the situations that cause Compliant power saving states to be entered.

**Management: FPT_PWR_EXT.1**

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

- Enable or disable the use of individual power saving states
- Specify one or more power saving state configurations

**Audit: FPT_PWR_EXT.1**

1 There are no auditable events foreseen.

2 **Management: FPT_PWR_EXT.2**

3 There are no management activities foreseen.

4 **Audit: FPT_PWR_EXT.2**

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

7 • Transition of the TSF into different power saving states

8 **FPT_PWR_EXT.1 Authorization Factor Acquisition**

9 Hierarchical to:       No other components

10 Dependencies:       No dependencies

11 **FPT_PWR_EXT.1.1** The TSF shall define the following Compliant power saving states:
12 [selection: choose at least one of: S3, S4, G2(S5), G3, [*assignment: other power saving*
13 *states*]].

14 **FPT_PWR_EXT.2 Authorization Factor Acquisition**

15 Hierarchical to:       No other components

16 Dependencies:       FPT_PWR_EXT.1 Power Saving States

17 **FPT_PWR_EXT.2.1** For each Compliant power saving state defined in
18 FPT_PWR_EXT.1.1, the TSF shall enter the Compliant power saving state when the
19 following conditions occur: [selection: choose at least one of: user-initiated request, system
20 shutdown, user inactivity, request initiated by remote management system, [*assignment:*
21 *other conditions*], no other conditions].

22 *FPT_TST_EXT TSF Testing*

23 **Family Behavior**

24 Components in this family address the requirements for self-testing the TSF for selected
25 correct operation.

26 **Component Leveling**

27

| FPT_TST_EXT TSF Testing | 1 |
|---|---|

1
2 FPT_TST_EXT.1, TSF Testing, requires a suite of self-tests to be run during initial start-up in order to demonstrate correct operation of the TSF.

3 **Management: FPT_TST_EXT.1**

4 No specific management functions are identified.

5 **Audit: FPT_TST_EXT.1**

6
7 The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

8 • Indication that TSF self-test was completed

9 **FPT_TST_EXT.1 TSF Testing**

10 Hierarchical to: No other components

11 Dependencies: No other components

12 **FPT_TST_EXT.1.1** The TSF shall run a suite of the following self-tests [selection: during
13 initial start-up (on power on), periodically during normal operation, at the request of the
14 authorized user, at the conditions [*assignment: conditions under which self-tests should*
15 *occur*]] to demonstrate the correct operation of the TSF: [*assignment: list of self-tests run by*
16 *the TSF*].

17 *FPT_TUD_EXT Trusted Update*

18 **Family Behavior**

19
20 Components in this family address the requirements for updating the TOE firmware and/or software.

21 **Component Leveling**

22

```
┌──────────────────────────────┐          ┌─────┐
│ FPT_TUD_EXT Trusted Update   │──────────│  1  │
└──────────────────────────────┘          └─────┘
```

23
24 FPT_TUD_EXT.1, Trusted Update, requires the capability to be provided to update the TOE firmware and software, including the ability to verify the updates prior to installation.

25 **Management: FPT_TUD_EXT.1**

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

27 • Ability to update the TOE and to verify the updates

1 **Audit: FPT_TUD_EXT.1**

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

4 • Initiation of the update process.
5 • Any failure to verify the integrity of the update

6 **FPT_TUD_EXT.1 Trusted Update**

7 Hierarchical to: No other components

8 Dependencies: FCS_COP.1(a) Cryptographic Operation (Signature Verification),
9 FCS_COP.1(b) Cryptographic Operation (Hash Algorithm)

10 **FPT_TUD_EXT.1.1** The TSF shall provide [*assignment: list of subjects*] the ability to query
11 the current version of the TOE software/firmware.

12 **FPT_TUD_EXT.1.2** The TSF shall provide [*assignment: list of subjects*] the ability to
13 initiate updates to TOE software/firmware.

14 **FPT_TUD_EXT.1.3** The TSF shall verify updates to the TOE software/firmware using a
15 [selection: digital signature, published hash] by the manufacturer prior to installing those
16 updates.

# 1 Appendix D: Entropy Documentation and Assessment

2 *This is an optional appendix in the cPP, and only applies if the TOE is providing*
3 *deterministic random bit generation services, e.g. the ST claims FCS_RBG_EXT.1.*

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

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

## 11 D.1 Design Description

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

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

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

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

## 29 D.2 Entropy Justification

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

1 The amount of information necessary to justify the expected min-entropy rate depends on the
2 type of entropy source included in the product.
3
4 For developer provided entropy sources, in order to justify the min-entropy rate, it is expected
5 that a large number of raw source bits will be collected, statistical tests will be performed,
6 and the min-entropy rate determined from the statistical tests. While no particular statistical
7 tests are required at this time, it is expected that some testing is necessary in order to
8 determine the amount of min-entropy in each output.
9
10 For third party provided entropy sources, in which the TOE vendor has limited access to the
11 design and raw entropy data of the source, the documentation will indicate an estimate of the
12 amount of min-entropy obtained from this third-party source. It is acceptable for the vendor
13 to "assume" an amount of min-entropy, however, this assumption must be clearly stated in
14 the documentation provided. In particular, the min-entropy estimate must be specified and the
15 assumption included in the ST.

16 Regardless of type of entropy source, the justification will also include how the DRBG is
17 initialized with the entropy stated in the ST, for example by verifying that the min-entropy
18 rate is multiplied by the amount of source data used to seed the DRBG or that the rate of
19 entropy expected based on the amount of source data is explicitly stated and compared to the
20 statistical rate. If the amount of source data used to seed the DRBG is not clear or the
21 calculated rate is not explicitly related to the seed, the documentation will not be considered
22 complete.
23
24 The entropy justification shall not include any data added from any third-party application or
25 from any state saving between restarts.

## D.3 Operating Conditions

27 The entropy rate may be affected by conditions outside the control of the entropy source
28 itself. For example, voltage, frequency, temperature, and elapsed time after power-on are just
29 a few of the factors that may affect the operation of the entropy source. As such,
30 documentation will also include the range of operating conditions under which the entropy
31 source is expected to generate random data. Similarly, documentation shall describe the
32 conditions under which the entropy source is no longer guaranteed to provide sufficient
33 entropy. Methods used to detect failure or degradation of the source shall be included.

## D.4 Health Testing

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

# 1 Appendix E: Key Management Description

2  The documentation of the product's encryption key management should be detailed enough
3  that, after reading, the evaluator will thoroughly understand the product's key management
4  and how it meets the requirements to ensure the keys are adequately protected. This
5  documentation should include an essay and diagram(s). This documentation is not required to
6  be part of the TSS - it can be submitted as a separate document and marked as developer
7  proprietary.

8  Essay:

9  The essay will provide the following information for all keys in the key chain:

10  • The purpose of the key
11  • If the key is stored in non-volatile memory
12  • How and when the key is protected
13  • How and when the key is derived
14  • The strength of the key
15  • When or if the key would be no longer needed, along with a justification.

16  The essay will also describe the following topics:

17  • A description of all authorization factors that are supported by the product and how
18    each factor is handled, including any conditioning and combining performed.
19  • If validation is supported, the process for validation shall be described, noting what
20    value is used for validation and the process used to perform the validation. It shall
21    describe how this process ensures no keys in the key chain are weakened or exposed
22    by this process.
23  • The authorization process that leads to the ultimate release of the BEV. This section
24    shall detail the key chain used by the product. It shall describe which keys are used in
25    the protection of the BEV and how they meet the derivation, key wrap, or a
26    combination of the two requirements, including the direct chain from the initial
27    authorization to the BEV. It shall also include any values that add into that key chain
28    or interact with the key chain and the protections that ensure those values do not
29    weaken or expose the overall strength of the key chain.
30  • The diagram and essay will clearly illustrate the key hierarchy to ensure that at no
31    point the chain could be broken without a cryptographic exhaust or all of the initial
32    authorization values and the effective strength of the BEV is maintained throughout
33    the Key Chain.
34  • A description of the data encryption engine, its components, and details about its
35    implementation (e.g. for hardware: integrated within the device's main SOC or
36    separate co-processor, for software: initialization of the product, drivers, libraries (if
37    applicable), logical interfaces for encryption/decryption, and areas which are not
38    encrypted (e.g. boot loaders, portions associated with the Master Boot Record

1       (MBRs), partition tables, etc.)). The description should also include the data flow
2       from the device's host interface to the device's persistent media storing the data,
3       information on those conditions in which the data bypasses the data encryption engine
4       (e.g. read-write operations to an unencrypted Master Boot Record area). The
5       description should be detailed enough to verify all platforms to ensure that when the
6       user enables encryption, the product encrypts all hard storage devices. It should also
7       describe the platform's boot initialization, the encryption initialization process, and at
8       what moment the product enables the encryption.

9     • The process for destroying keys when they are no longer needed by describing the
10       storage location of all keys and the protection of all keys stored in non-volatile
11       memory.

12 Diagram:

13     • The diagram will include all keys from the initial authorization factor(s) to the BEV
14       and any keys or values that contribute into the chain. It must list the cryptographic
15       strength of each key and indicate how each key along the chain is protected with
16       either Key Derivation or Key Wrapping (from the allowed options). The diagram
17       should indicate the input used to derive or unwrap each key in the chain.

18     • A functional (block) diagram showing the main components (such as memories and
19       processors) and the data path between, for hardware, the device's host interface and
20       the device's persistent media storing the data, or for software, the initial steps needed
21       for the activities the TOE performs to ensure it encrypts the storage device entirely
22       when a user or administrator first provisions the product. The hardware encryption
23       diagram shall show the location of the data encryption engine within the data path.

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

# 1    Appendix F: Glossary

| Term | Meaning |
|---|---|
| **Authorization Factor** | A value that a user knows, has, or is (e.g. password, token, etc.) submitted to the TOE to establish that the user is in the community authorized to use the hard disk. This value is used in the derivation or decryption of the BEV and eventual decryption of the DEK. Note that these values may or may not be used to establish the particular identity of the user. |
| **Assurance** | Grounds for confidence that a TOE meets the SFRs [CC1]. |
| **Border Encryption Value** | A value passed from the AA to the EE intended to link the key chains of the two components. |
| **Key Sanitization** | A method of sanitizing encrypted data by securely overwriting the key that was encrypting the data. |
| **Data Encryption Key (DEK)** | A key used to encrypt data-at-rest. |
| **Full Drive Encryption** | Refers to partitions of logical blocks of user accessible data as managed by the host system that indexes and partitions and an operating system that maps authorization to read or write data to blocks in these partitions. For the sake of this Security Program Definition (SPD) and cPP, FDE performs encryption and authorization on one partition, so defined and supported by the OS and file system jointly, under consideration. FDE products encrypt all data (with certain exceptions) on the partition of the storage device and permits access to the data only after successful authorization to the FDE solution. The exceptions include the necessity to leave a portion of the storage device (the size may vary based on implementation) unencrypted for such things as the Master Boot Record (MBR) or other AA/EE pre-authentication software. These FDE cPPs interpret the term "full drive encryption" to allow FDE solutions to leave a portion of the storage device unencrypted so long as it contains no protected data. |
| **Intermediate Key** | A key used in a point between the initial user authorization and the DEK. |
| **Host Platform** | The local hardware and software the TOE is running on, and does not include any peripheral devices (e.g. USB devices) that may be connected to the local hardware and software. |
| **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. |
| **Key Material** | Key material is commonly known as critical security parameter (CSP) data, and also includes authorization data, nonces, and metadata. |
| **Key Release Key (KRK)** | A key used to release another key from storage, it is not used for the direct derivation or decryption of another key. |

| Term | Meaning |
|---|---|
| **Operating System (OS)** | Software which runs at the highest privilege level and can directly control hardware resources. |
| **Non-Volatile Memory** | A type of computer memory that will retain information without power. |
| **Powered-Off State** | The device has been shut down. |
| **Protected Data** | This refers to all data on the storage device with the exception of a small portion required for the TOE to function correctly. It is all space on the disk a user could write data to and includes the operating system, applications, and user data. Protected data does not include the Master Boot Record or Pre-authentication area of the drive – areas of the drive that are necessarily unencrypted. |
| **Submask** | A submask is a bit string that can be generated and stored in a number of ways. |
| **Target of Evaluation** | A set of software, firmware and/or hardware possibly accompanied by guidance. [CC1] |

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

1 **Appendix G: Acronyms**

| Acronym | Meaning |
|---|---|
| AA | Authorization Acquisition |
| AES | Advanced Encryption Standard |
| BEV | Border Encryption Value |
| BIOS | Basic Input Output System |
| CBC | Cipher Block Chaining |
| CC | Common Criteria |
| CCM | Counter with CBC-Message Authentication Code |
| CEM | Common Evaluation Methodology |
| CPP | Collaborative Protection Profile |
| DEK | Data Encryption Key |
| DRBG | Deterministic Random Bit Generator |
| DSS | Digital Signature Standard |
| ECC | Elliptic Curve Cryptography |
| ECDSA | Elliptic Curve Digital Signature Algorithm |
| EE | Encryption Engine |
| EEPROM | Electrically Erasable Programmable Read-Only Memory |
| FIPS | Federal Information Processing Standards |
| FDE | Full Drive Encryption |
| FFC | Finite Field Cryptography |
| GCM | Galois Counter Mode |
| HMAC | Keyed-Hash Message Authentication Code |
| HW | Hardware |
| IEEE | Institute of Electrical and Electronics Engineers |
| IT | Information Technology |
| ITSEF | IT Security Evaluation Facility |
| ISO/IEC | International Organization for Standardization / International Electrotechnical Commission |
| IV | Initialization Vector |
| KEK | Key Encryption Key |
| KMD | Key Management Description |
| KRK | Key Release Key |
| MBR | Master Boot Record |
| NIST | National Institute of Standards and Technology |
| OS | Operating System |
| PBKDF | Password-Based Key Derivation Function |
| PRF | Pseudo Random Function |
| RBG | Random Bit Generator |
| RNG | Random Number Generator |
| RSA | Rivest Shamir Adleman Algorithm |
| SAR | Security Assurance Requirements |
| SED | Self Encrypting Drive |
| SHA | Secure Hash Algorithm |
| SFR | Security Functional Requirements |
| ST | Security Target |
| SPD | Security Problem Definition |

| Acronym | Meaning |
|---------|---------|
| SPI | Serial Peripheral Interface |
| TOE | Target of Evaluation |
| TPM | Trusted Platform Module |
| TSF | TOE Security Functionality |
| TSS | TOE Summary Specification |
| USB | Universal Serial Bus |
| XOR | Exclusive or |
| XTS | XEX (XOR Encrypt XOR) Tweakable Block Cipher with Ciphertext Stealing |

1

# Appendix H: References

National Institute of Standards and Technology (NIST) Special Publication 800-38F, Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping, National Institute of Standards and Technology, December 2012.

National Institute of Standards and Technology (NIST) Special Publication 800-56B, Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography, National Institute of Standards and Technology, August 2009.

National Institute of Standards and Technology (NIST) Special Publication 800-88 Revision 1, Guidelines for Media Sanitization, National Institute of Standards and Technology, December 2014.

National Institute of Standards and Technology (NIST) Special Publication 800-90A, Recommendation for Random Number Generation Using Deterministic Random Bit Generators, National Institute of Standards and Technology, January 2012.

National Institute of Standards and Technology (NIST) Special Publication 800-132, Recommendation for Password-Based Key Derivation Part 1: Storage Applications, National Institute of Standards and Technology, December 2010.

Federal Information Processing Standard Publication (FIPS-PUB) 186-4, Digital Signature Standard (DSS), National Institute of Standards and Technology, July 2013.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 9796-2:2010 (3rd edition), Information technology — Security techniques — Digital signature schemes giving message recovery, International Organization for Standardization/International Electrotechnical Commission, 2010.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 9797-2:2011 (2nd edition), Information technology — Security techniques — Message Authentication Codes (MACs) – Part 2: Mechanisms using a dedicated hash-function, International Organization for Standardization/International Electrotechnical Commission, 2011.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 10116:2006 (3rd edition), Information technology — Security techniques — Modes of operation for an n-bit block cipher, International Organization for Standardization/International Electrotechnical Commission, 2006.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 10118-3:2004 (3rd edition), Information technology — Security techniques — Hash-functions – Part 3: Dedicated hash-functions, International Organization for Standardization/International Electrotechnical Commission, 2004.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 14888-3:2006 (2nd edition), Information technology — Security techniques —Digital signatures with appendix – Part 3: Discrete logarithm based mechanisms, International Organization for Standardization/International Electrotechnical Commission, 2006.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 18031:2011 (2nd edition), Information technology — Security techniques — Random bit generation, International Organization for Standardization/International Electrotechnical Commission, 2011.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 18033-3:2011 (3rd edition), Information technology — Security techniques — Encryption algorithms – Part 3: Block ciphers, International Organization for Standardization/International Electrotechnical Commission, 2011.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 19772:2009, Information technology — Security techniques Authenticated encryption, International Organization for Standardization/International Electrotechnical Commission, 2009.