

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

3 February 1, 2019

1 **Acknowledgements**

- 2 This collaborative Protection Profile (cPP) was developed by the Full Drive Encryption
- 3 international Technical Community with representatives from industry, Government agencies,
- 4 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 captured
7 within this cPP are described in the *Supporting Document (Mandatory Technical Document)*
8 *Full Drive Encryption: Authorization Acquisition, February 2019*.

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 defines
26 the IT security requirements of a technology specific type of TOE and specifies the functional
27 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, evaluators
30 and schemes.

1 **0.4 Related Documents**

2 **Protection Profiles**

3 [FDE – EE] collaborative Protection Profile for Full Drive Encryption – Encryption Engine,
4 Version 2.0 + Errata 20190201, February 1, 2019

5 **Common Criteria¹**

[CC1] Common Criteria for Information Technology Security Evaluation,
Part 1: Introduction and General Model,
CCMB-2017-04-001, Version 3.1 Revision 5, April 2017.

[CC2] Common Criteria for Information Technology Security Evaluation,
Part 2: Security Functional Components,
CCMB-2017-04-002, Version 3.1 Revision 5, April 2017.

[CC3] Common Criteria for Information Technology Security Evaluation,
Part 3: Security Assurance Components,
CCMB-2017-04-003, Version 3.1 Revision 5, April 2017.

[CEM] Common Methodology for Information Technology Security Evaluation,
Evaluation Methodology,
CCMB-2017-04-004, Version 3.1, Revision 5, April 2017.

[SD] Supporting Document (Mandatory Technical Document), Full Drive
Encryption: Authorization Acquisition, February 2019

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.
2.0 + Errata 20190201	February 1, 2019	Updated to reflect CC Part 3 evaluation findings and FDE Interpretation Team [FIT] rulings

2

1	Contents	
2	Acknowledgements	2
3	0. Preface	3
4	0.1 Objectives of Document	3
5	0.2 Scope of Document.....	3
6	0.3 Intended Readership	3
7	0.4 Related Documents.....	4
8	Protection Profiles.....	4
9	Common Criteria	4
10	0.5 Revision History	5
11	1. PP Introduction	10
12	1.1 PP Reference Identification	10
13	1.2 Introduction to the FDE Collaborative Protection Profiles (cPPs) Effort	10
14	1.3 Implementations	11
15	1.4 Target of Evaluation (TOE) Overview	11
16	1.4.1 Authorization Acquisition Introduction	12
17	1.4.2 Authorization Acquisition Security Capabilities.....	13
18	1.4.3 Interface/Boundary.....	13
19	1.5 The TOE and the Operational/Pre-Boot Environments	13
20	1.6 TOE Use Case	14
21	2. CC Conformance Claims	15
22	3. Security Problem Definition	16
23	3.1 Threats	16
24	3.2 Assumptions	19
25	3.3 Organizational Security Policy	21
26	4. Security Objectives	22
27	4.1 Security Objectives for the Operational Environment.....	22
28	5. Security Functional Requirements	24
29	5.1 Conventions	24
30	5.2 SFR Architecture	25
31	5.3 Class: Cryptographic Support (FCS).....	25
32	FCS_AFA_EXT.1 Authorization Factor Acquisition.....	25
33	FCS_AFA_EXT.2 Timing of Authorization Factor Acquisition.....	26
34	FCS_CKM.4(a) Cryptographic Key Destruction (Power Management).....	26
35	FCS_CKM.4(d) Cryptographic Key Destruction (Software TOE, 3 rd Party Storage).....	26
36	FCS_CKM_EXT.4(a) Cryptographic Key and Key Material Destruction (Destruction Timing)	27
37	FCS_CKM_EXT.4(b) Cryptographic Key and Key Material Destruction (Power Management)	28
38	FCS_KYC_EXT.1 Key Chaining (Initiator)	28
39	FCS_SNI_EXT.1 Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation).....	29
40	5.4 Class: Security Management (FMT).....	30
41	FMT_MOF.1 Management of Functions Behavior	30
42	FMT_SMF.1 Specification of Management Functions	30
43	FMT_SMR.1 Security Roles	31
44	5.5 Class: Protection of the TSF (FPT).....	31
45	FPT_KYP_EXT.1 Protection of Key and Key Material.....	31
46	FPT_PWR_EXT.1 Power Saving States	31
47	FPT_PWR_EXT.2 Timing of Power Saving States	32
48	FPT_TUD_EXT.1 Trusted Update.....	32
49	6. Security Assurance Requirements.....	33
50	6.1 ASE: Security Target.....	33
51	6.2 ADV: Development	34
52	6.2.1 Basic Functional Specification (ADV_FSP.1).....	34

1	6.3	AGD: Guidance Documentation.....	34
2	6.3.1	Operational User Guidance (AGD_OPE.1)	35
3	6.3.2	Preparative Procedures (AGD_PRE.1)	35
4	6.4	Class ALC: Life-cycle Support.....	35
5	6.4.1	Labelling of the TOE (ALC_CMC.1)	35
6	6.4.2	TOE CM Coverage (ALC_CMS.1).....	35
7	6.5	Class ATE: Tests	35
8	6.5.1	Independent Testing – Conformance (ATE_IND.1)	36
9	6.6	Class AVA: Vulnerability Assessment.....	36
10	6.6.1	Vulnerability Survey (AVA_VAN.1)	36
11		Appendix A: Optional Requirements	37
12	A.1	Internal Cryptographic Implementation.....	37
13	A.2	TSF Self-Testing.....	37
14		FPT_TST_EXT.1 TSF Testing.....	38
15		Appendix B: Selection-Based Requirements.....	39
16	B.1	Class: Cryptographic Support (FCS).....	39
17		FCS_CKM.1(a) Cryptographic Key Generation (Asymmetric Keys)	39
18		FCS_CKM.1(b) Cryptographic Key Generation (Symmetric Keys)	40
19		FCS_COP.1(a) Cryptographic Operation (Signature Verification)	40
20		FCS_COP.1(b) Cryptographic Operation (Hash Algorithm).....	41
21		FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm)	41
22		FCS_COP.1(d) Cryptographic Operation (Key Wrapping).....	41
23		FCS_COP.1(e) Cryptographic Operation (Key Transport)	42
24		FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption)	42
25		FCS_COP.1(g) Cryptographic Operation (Key Encryption)	42
26		FCS_KDF_EXT.1 Cryptographic Key Derivation	43
27		FCS_PCC_EXT.1 Cryptographic Password Construct and Conditioning.....	43
28		FCS_RBG_EXT.1 Cryptographic Operation (Random Bit Generation).....	43
29		FCS_SMC_EXT.1 Submask Combining.....	44
30		FCS_VAL_EXT.1 Validation	44
31		Appendix C: Extended Component Definitions	46
32	C.1	Background and Scope	46
33	C.2	Extended Component Definitions	46
34		FCS_AFA_EXT Authorization Factor Acquisition.....	46
35		FCS_CKM_EXT Cryptographic Key Management	48
36		FCS_KDF_EXT Cryptographic Key Derivation.....	49
37		FCS_KYC_EXT Key Chaining.....	50
38		FCS_PCC_EXT Cryptographic Password Construction and Conditioning.....	52
39		FCS_RBG_EXT Random Bit Generation	53
40		FCS_SMC_EXT Submask Combining.....	54
41		FCS_SNI_EXT Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation).....	55
42		FPT_KYP_EXT Key and Key Material Protection	57
43		FPT_PWR_EXT Power Management	58
44		FPT_TST_EXT TSF Testing.....	60
45		FPT_TUD_EXT Trusted Update.....	61
46		Appendix D: Entropy Documentation and Assessment.....	63
47	D.1	Design Description.....	63
48	D.2	Entropy Justification	63
49	D.3	Operating Conditions	64
50	D.4	Health Testing	64
51		Appendix E: Key Management Description	65
52		Appendix F: Glossary.....	67

1	Appendix G: Acronyms.....	69
2	Appendix H: References.....	71
3		

1	Figures / Tables	
2	Figure 1: FDE components	10
3	Table 1: Examples of cPP Implementations	11
4	Figure 2: Authorization Acquisition Details	12
5	Figure 3: Operational Environment	14
6	Table 2: TOE Security Functional Requirements	25
7	Table 3: TOE Security Assurance Requirements	33
8	Table 4: Extended Components	46
9		

1. PP Introduction

1.1 PP Reference Identification

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

PP Version: 2.0 + Errata 20190201

PP Date: February 1, 2019

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

27

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 piece
 2 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, and
 7 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 evaluate
 14 against the applicable cPP. The FDE cPP is divided into two documents to allow labs to
 15 independently evaluate solutions tailored to one cPP or the other. When a customer acquires
 16 an FDE solution, they will either obtain a single vendor product that meets the AA + EE cPPs
 17 or two products, one of which meets the AA and the other of which meets the EE cPPs.

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

19 *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

20 **1.4 Target of Evaluation (TOE) Overview**

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

3 1.4.1 Authorization Acquisition Introduction

4 The Authorization Acquisition sends a Border Encryption Value (BEV), which could be a Key
 5 Encryption Key (KEK), a Key Releasing Key (KRR), 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 eventually
 8 protect the DEK. A KEK wraps other keys, notably the DEK or other intermediary keys which

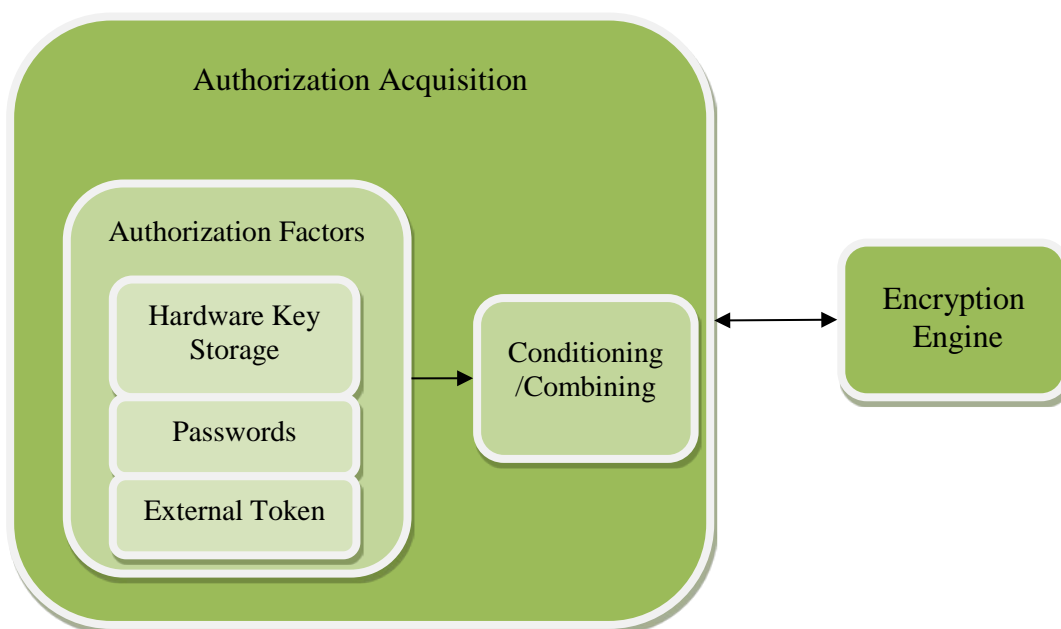


Figure 2: Authorization Acquisition Details

9 chain to the DEK. Key Releasing Keys (KRRs) authorize the EE to release either the DEK or
 10 other intermediary keys which chain to the DEK. Figure 2 illustrates the components within
 11 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 that
 14 the possessor of the authorization factor belongs to the community of users authorized to access
 15 information stored on the storage device (and does not require specific user authorization).
 16 Examples of authorization factors include, but are not limited to, passwords, passphrases, or
 17 randomly generated values stored on USB tokens or a pin to release a key on hardware storage
 18 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 factor,
4 the AA may condition them further. For example, it may apply an approved password-based
5 key derivation function (e.g. PBKDF2) on passwords. An external token containing a randomly
6 generated value of sufficient strength may require no further conditioning on the authorization
7 factors. The AA may then combine one or more authorization factors in such a way that
8 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 handle
11 these features. The management functionality may include the ability to send commands to the
12 EE such as changing a DEK, setting up new users, managing KEKs and other intermediate
13 keys, and performing a key sanitization (e.g. overwrite of the DEK). It may also forward
14 commands that partition the drive for use by multiple users. However, this document defers the
15 management of partitions and assumes that administrators will only provision and manage the
16 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 implementation.
19 If one vendor provides the entire FDE solution, then it may choose to not implement an
20 interface between the AA and EE components. If a vendor provides a solution for one of the
21 components, then the assumptions below state that the channel between the two components is
22 sufficiently secure. Although standards and specifications exist for the interface between AA
23 and EE components, the cPP does not require vendors to follow the standards in this version.

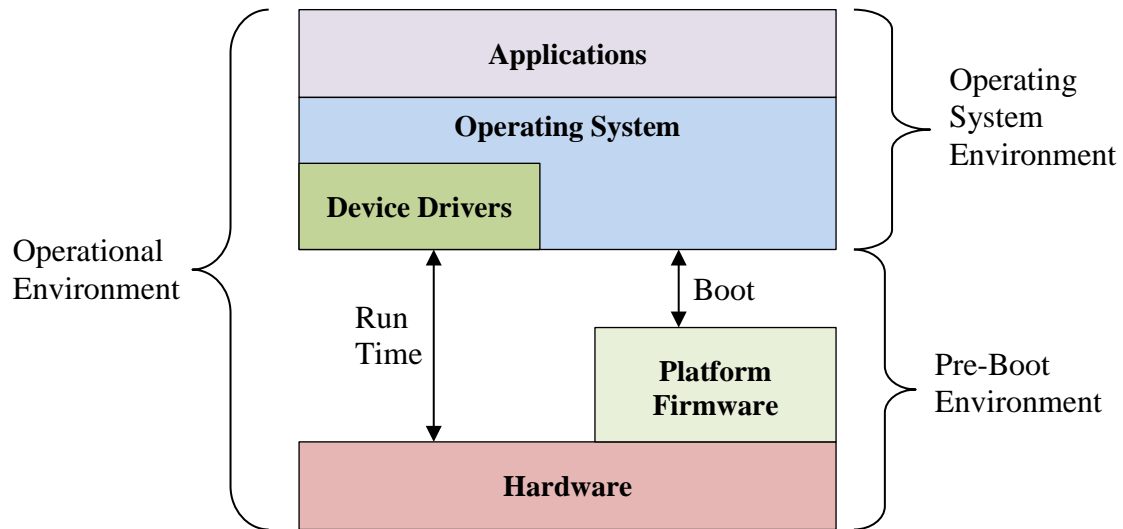
24 **1.5 The TOE and the Operational/Pre-Boot Environments**

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

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

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

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



3 *Figure 3: Operational Environment*

4 **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 modifications
 8 to the environment or the TOE itself (e.g., evil maid attacks) is not addressed by these cPPs
 9 (i.e., FDE-AA and FDE- EE).

2. CC Conformance Claims

- 1
- 2 As defined by the references [CC1], [CC2], and [CC3], this cPP conforms to the requirements
3 of Common Criteria v3.1, Release 5. This cPP is CC Part 2 extended and CC Part 3 conformant.
4 Extended component definitions can be found in Appendix C.
- 5 The methodology applied for the cPP evaluation is defined in [CEM].
- 6 This cPP satisfies the following Assurance Families: APE_CCL.1, APE_ECD.1, APE_INT.1,
7 APE_OBJ.1, APE_REQ.1 and APE_SPD.1.
- 8 This cPP does not claim conformance to another PP.
- 9 In order to be conformant to this cPP, a TOE must demonstrate *Exact Conformance*. *Exact*
10 *Conformance* is defined as the ST containing all of the requirements in section 5 of this cPP,
11 and potentially requirements from Appendix A or Appendix B of this cPP. While iteration is
12 allowed, no additional requirements (from the CC parts 2 or 3) are allowed to be included in
13 the ST. Further, no requirements in section 5 of this cPP are allowed to be omitted.
- 14 An ST can claim conformance with the FDE-EE in combination with the FDE-AA and still
15 maintain exact conformance.
- 16 The FDE Enterprise Management (EM) PP-Module can specify the FDE-AA as a Base-PP for
17 use with the FDE-EM in a PP-configuration and can also specify the FDE-AA and FDE-EE as
18 a set of Base-PPs for use with the FDE-EM in a PP-configuration.

3. Security Problem Definition

3.1 Threats

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

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

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

(T.UNAUTHORIZED_DATA_ACCESS) The cPP addresses the primary threat of unauthorized disclosure of protected data stored on a storage device. If an adversary obtains a lost or stolen storage device (e.g., a storage device contained in a laptop or a portable external storage device), they may attempt to connect a targeted storage device to a host of which they

1 have complete control and have raw access to the storage device (e.g., to specified disk sectors,
2 to specified blocks).

3 [FCS_AFA_EXT.2, FMT_MOF.1, FMT_SMF.1, FMT_SMR.1, FPT_PWR_EXT.1,
4 FPT_PWR_EXT.2, FCS_VAL_EXT.1, FPT_TST_EXT.1]

5 Rationale: [FCS_AFA_EXT.2] requires authentication to be re-entered upon return from
6 a compliant power state. [FMT_MOF.1] restricts the ability to modify compliant power
7 states to authorized users defined by [FMT_SMR.1]. [FPT_PWR_EXT.1] defines what
8 power states are compliant for the TOE. [FPT_PWR_EXT.2] defines conditions in which
9 the TOE will enter a compliant power state. These requirements ensure the device is
10 secure if lost in a compliant power state.

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

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

22 [FCS_AFA_EXT.1, FCS_AFA_EXT.2, FCS_CKM.4(a), FCS_CKM.4(b),
23 FCS_CKM_EXT.4(a), FCS_CKM_EXT.4(b), FCS_KYC_EXT.1, FMT_MOF.1,
24 FMT_SMF.1, FMT_SMR.1, FPT_KYP_EXT.1, FPT_PWR_EXT.1,
25 FPT_PWR_EXT.2, FCS_SNI_EXT.1, FCS_VAL_EXT.1, FPT_TST_EXT.1,
26 FCS_CKM.1(a), FCS_CKM.1(b), FCS_COP.1(b), FCS_COP.1(c), FCS_COP.1(d),
27 FCS_COP.1(e), FCS_COP.1(f), FCS_COP.1(g), FCS_KDF_EXT.1, FCS_PCC_EXT.1,
28 FCS_RBG_EXT.1, FCS_SMC_EXT.1]

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

- 34 • Key derivation [FCS_KDF_EXT.1]
- 35 • Key wrapping [FCS_COP.1(d)]
- 36 • Key combining [FCS_SMC_EXT.1]
- 37 • Key transport [FCS_COP.1(e)]
- 38 • Key encryption [FCS_COP.1(g)]

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

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

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

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

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

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

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

34 [FCS_AFA_EXT.1] requires a set of authorization factors that will be difficult to guess,
35 user provides factors are conditioned [FCS_PCC_EXT.1] to increase the cost of
36 repeatedly guessing a user provided value. [FCS_SNI_EXT.1] requires proper salts,
37 which will prevent pre-computed attacks. FCS_SMC_EXT.1 allows for multifactor
38 authentication options, further increasing the difficulty of guessing a authentication
39 factor.

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

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

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

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

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

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

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

21 **3.2 Assumptions**

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

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

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

33 [OE.INITIAL_DRIVE_STATE]

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

1 [OE.POWER_DOWN]

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

10 [OE.TRUSTED_CHANNEL]

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

14 [OE.PASSPHRASE_STRENGTH, OE.POWER_DOWN, OE.SINGLE_USE_ET,
15 OE.TRAINED_USERS]

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

19 [OE.PLATFORM_STATE]

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

22 [OE.SINGLE_USE_ET]

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

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

28 [OE.POWER_DOWN]

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

32 [OE.PASSPHRASE_STRENGTH]

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

5 [OE.PLATFORM_I&A]

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

9 [OE.STRONG_ENVIRONMENT_CRYPTO]

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

13 [OE.PHYSICAL]

14 **3.3 Organizational Security Policy**

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

4. Security Objectives

4.1 Security Objectives for the Operational Environment

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

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

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

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

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

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

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

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

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

(OE.SINGLE_USE_ET) External tokens that contain authorization factors will be used for 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 for
9 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 the
12 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 connected)
14 is free of malware that could interfere with the correct operation of the product.

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

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

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

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

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

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

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

34

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—when the refinement operation substitutes text, only the added text is included;
- Selection: Indicated with underlined text;
- Assignment within a Selection: Indicated with *italicized and underlined text*;
- 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, 3 rd 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)
Security Management (FMT)	FCS_SNI_EXT.1 Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation)
	FMT_MOF.1 Management of Functions Behavior
	FMT_SMF.1 Specification of Management Functions
Protection of the TSF (FPT)	FMT_SMR.1 Security Roles
	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].

1 **Application Note:** This requirement specifies what authorization factors the TOE accepts from
2 the user. A password entered by the user is one authorization factor that the TOE must be able
3 to condition, as specified in FCS_PCC_EXT.1. Another option is a smartcard authorization
4 factor, with the differentiating feature being how the value is generated – either by the TOE’s
5 RBG or by the platform. An external USB token may also be used, with the submask value
6 generated either by the TOE’s RBG or by the platform.

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

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

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

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

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

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

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

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

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

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

- 30 • **For volatile memory, the destruction shall be executed by a [selection:**
 - 31 ○ **single overwrite consisting of [selection:**
 - 32 ▪ **a pseudo-random pattern using the TSF’s RBG,**
 - 33 ▪ **zeroes,**
 - 34 ▪ **ones,**
 - 35 ▪ **a new value of a key,**

- 1 ▪ [assignment: some value that does not contain any CSP],
- 2 ○ removal of power to the memory,
- 3 ○ destruction of reference to the key directly followed by a request for garbage
- 4 collection];
- 5 • For non-volatile storage that consists of the invocation of an interface provided by
- 6 the underlying platform that [selection:
- 7 ○ logically addresses the storage location of the key and performs a [selection:
- 8 single, [assignment: ST author defined multi-pass]] overwrite consisting of
- 9 [selection:
- 10 ▪ a pseudo-random pattern using the TSF's RBG,
- 11 ▪ zeroes,
- 12 ▪ ones,
- 13 ▪ a new value of a key,
- 14 ▪ [assignment: some value that does not contain any CSP];
- 15 ○ instructs the underlying platform to destroy the abstraction that represents the
- 16 key]
- 17]]

18 that meets the following: [no standard].

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

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

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

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

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

38 **Application Note:** Keys, including intermediate keys and key material that are no longer
39 needed are destroyed by using an approved method, FCS_CKM.4(d). Examples of keys are

1 *intermediate keys, submasks, and BEV. There may be instances where keys or key material that*
2 *are contained in persistent storage are no longer needed and require destruction. Based on*
3 *their implementation, vendors will explain when certain keys are no longer needed. There are*
4 *multiple situations in which key material is no longer necessary, for example, a wrapped key*
5 *may need to be destroyed when a password is changed. However, there are instances when*
6 *keys are allowed to remain in memory, for example, a device identification key. If a PIN was*
7 *used for a smartcard, the TSF should ensure that the PIN was properly destroyed.*

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

- 25
- 26 • CBC: IVs shall be non-repeating and unpredictable;
 - 27 • CCM: Nonce shall be non-repeating and unpredictable;
 - 28 • XTS: No IV. Tweak values shall be non-negative integers, assigned consecutively,
29 and starting at an arbitrary non-negative integer;
 - 30 • GCM: IV shall be non-repeating. The number of invocations of GCM shall not exceed
2³² for a given secret key].

31 **Application Note:** This requirement covers several important factors – the salt must be
32 random, but the nonces only have to be unique. FCS_SNI_EXT.1.3 specifies how the IV
33 should be handled for each encryption mode. CBC, XTS, and GCM are allowed for AES
34 encryption of the data. AES-CCM is an allowed mode for Key Wrapping.

1 **5.4 Class: Security Management (FMT)**

2 **FMT_MOF.1 Management of Functions Behavior**

3 **FMT_MOF.1.1** The TSF shall restrict the ability to [modify the behaviour of] the functions
4 [*use of Compliant power saving state*] to [*authorized users*].

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

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

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

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

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

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

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

1 **FMT_SMR.1 Security Roles**

2 **FMT_SMR.1.1** The TSF shall maintain the roles [*authorized user*].

3 **FMT_SMR.1.2** The TSF shall be able to associate users with roles.

4 **5.5 Class: Protection of the TSF (FPT)**

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

6 **FPT_KYP_EXT.1.1** The TSF shall [selection:

- 7 • not store keys in non-volatile memory
- 8 • only store keys in non-volatile memory when wrapped, as specified in
- 9 FCS_COP.1(d), or encrypted, as specified in FCS_COP.1(g) or FCS_COP.1(e)
- 10 • only store plaintext keys that meet any one of the following criteria [selection:
 - 11 ○ the plaintext key is not part of the key chain as specified in
 - 12 FCS_KYC_EXT.1,
 - 13 ○ the plaintext key will no longer provide access to the encrypted data after
 - 14 initial provisioning,
 - 15 ○ the plaintext key is a key split that is combined as specified in
 - 16 FCS_SMC_EXT.1, and the other half of the key split is [selection:
 - 17 ▪ wrapped as specified in FCS_COP.1(d),
 - 18 ▪ encrypted as specified in FCS_COP.1(g) or FCS_COP.1(e),
 - 19 ▪ derived and not stored in non-volatile memory],
 - 20 ○ the non-volatile memory the key is stored on is located in an external storage
 - 21 device for use as an authorization factor,
 - 22 ○ the plaintext key is [selection:
 - 23 ▪ used to wrap a key as specified in FCS_COP.1(d),
 - 24 ▪ used to encrypt a key as specified in FCS_COP.1(g) or FCS_COP.1(e)]
 - 25 that is already [selection:
 - 26 ▪ wrapped as specified in FCS_COP.1(d),
 - 27 ▪ encrypted as specified in FCS_COP.1(g) or FCS_COP.1(e)]].

28 **Application Note:** *The plaintext key storage in non-volatile memory is allowed for several*

29 *reasons. If the keys exist within protected memory that is not user accessible on the TOE or*

30 *OE, the only methods that allow it to play a security relevant role for protecting the BEV or*

31 *the DEK are if it is a key split or providing additional layers of wrapping or encryption on keys*

32 *that have already been protected.*

33 **FPT_PWR_EXT.1 Power Saving States**

34 **FPT_PWR_EXT.1.1** The TSF shall define the following Compliant power saving states:

35 [selection: choose at least one of: S3, S4, G2(S5), G3, [assignment: other power saving states]].

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

3 **FPT_PWR_EXT.2 Timing of Power Saving States**

4 **FPT_PWR_EXT.2.1** For each Compliant power saving state defined in FPT_PWR_EXT.1.1,
5 the TSF shall enter the Compliant power saving state when the following conditions occur:
6 user-initiated request, [selection: shutdown, user inactivity, request initiated by remote
7 management system, [assignment: other conditions], no other conditions].

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

12 **FPT_TUD_EXT.1 Trusted Update**

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

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

17 **FPT_TUD_EXT.1.3 Refinement:** The TSF shall verify updates to the TOE software using a
18 [**digital signature as specified in FCS COP.1(a)**] by the manufacturer prior to installing
19 those updates.

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

6. Security Assurance Requirements

This cPP identifies the Security Assurance Requirements (SARs) to frame the extent to which the evaluator assesses the documentation applicable for the evaluation and performs independent testing. Individual Evaluation Activities to be performed are specified in *Supporting Document (Mandatory Technical Document) Full Drive Encryption: Authorization Acquisition, February 2019*.

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

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

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)

6.1 ASE: Security Target

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

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

1 description of their key management implementation. This information can be submitted as an
2 appendix to the ST and marked proprietary, as this level of detailed information is not expected
3 to be made publicly available. See Appendix E for details on the expectation of the developer’s
4 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), and**
9 **[selection: Entropy Essay, list of all of 3rd party software libraries (including version**
10 **numbers), 3rd party hardware components (including model/version numbers), no other**
11 **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 not
18 necessary to have a formal or complete specification of these interfaces. Additionally, because
19 TOEs conforming to this cPP will may interfaces to the Operational Environment that are not
20 directly invoked by TOE users, there is little point specifying that such interfaces be described
21 in and of themselves since only indirect testing of such interfaces may be possible. For this
22 cPP, the Evaluation Activities for this family focus on understanding the interfaces presented
23 in the TSS in response to the functional requirements and the interfaces presented in the AGD
24 documentation. No additional “functional specification” documentation is necessary to satisfy
25 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 of
31 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 **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 integrators can be spread among documents or web pages.

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

12 **6.3.2 Preparative Procedures (AGD_PRE.1)**

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

15 **6.4 Class ALC: Life-cycle Support**

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

22 **6.4.1 Labelling of the TOE (ALC_CMC.1)**

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

26 **6.4.2 TOE CM Coverage (ALC_CMS.1)**

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

29 **6.5 Class ATE: Tests**

30 Testing is specified for functional aspects of the system as well as aspects that take advantage
31 of design or implementation weaknesses. The former is done through the ATE_IND family,
32 while the latter is through the AVA_VAN family. For this cPP, testing is based on advertised
33 functionality and interfaces with dependency on the availability of design information. One of

1 the primary outputs of the evaluation process is the test report as specified in the following
2 requirements.

3 **6.5.1 Independent Testing – Conformance (ATE_IND.1)**

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

11 **6.6 Class AVA: Vulnerability Assessment**

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

17 **6.6.1 Vulnerability Survey (AVA_VAN.1)**

18 Appendix A in the companion Supporting Document provides a guide to the evaluator in
19 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 other
4 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 have
6 to be in order for a TOE to claim conformance to this cPP. The second type (in Appendix B) is
7 requirements based on selections in the body of the cPP: if certain selections are made, then
8 additional requirements in that appendix will need to be included in the body of the ST (e.g.,
9 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 for
11 incorporating the appropriate requirements from the appendices into the body of their ST, the
12 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 that
16 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' SFRs
20 due to the fact that their functionality is contingent on the ST author make certain selections in
21 other SFRs. Because of this, these SFRs have been placed in Appendix B. Note however that
22 there is still an expectation that some of these functions may be provided by the Operational
23 Environment, in which case it is acceptable to omit the SFRs in question so long as the ST
24 author can provide evidence that the Operational Environment will include a cryptographic
25 interface to the TSF that allows for secure usage of these functions, and that the functions have
26 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 omit
29 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 FCS_RBG_EXT.1
35 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 should verify that the TSS describes health tests that are consistent with section*
9 *11.3 of NIST SP 800-90.*

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

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

1 Appendix B: Selection-Based Requirements

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

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

10 B.1 Class: Cryptographic Support (FCS)

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

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

17 [FCS_VAL_EXT.1]

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

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

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

- 24 • *RSA schemes using cryptographic key sizes of [selection: 2048-bit, 3072-bit, 4096-*
25 *bit] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)",*
26 *Appendix B.3;*
- 27 • *ECC schemes using "NIST curves" of [selection: P-256, P-384, P-521] that meet*
28 *the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix*
29 *B.4;*
- 30 • *FFC schemes using cryptographic key sizes of [selection: 2048-bit, 3072-bit, 4096-*
31 *bit] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)",*
32 *Appendix B.1*

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

3 **Application Note:** Asymmetric keys may be used to “wrap” a key or submask. This SFR should
4 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) generate
10 seeds for RSA and to b) generate private keys directly for ECC and FFC. So FCS_RBG_EXT.1
11 is used together with this SFR. A hash algorithm is also required when the key pair generation
12 algorithm is selected based on either Appendix B.3.2 or B.3.5 of FIPS 186-4. So in such case,
13 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 **using**
16 **a Random Bit Generator as specified in FCS_RBG_EXT.1** and specified cryptographic key
17 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 [selection: 2048-bit,**
24 **3072-bit, 4096-bit];**

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

26]

27 that meet the following: [~~selection:~~

28 • **FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 5.5, using PKCS #1**
29 **v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1-v1 5; ISO/IEC**
30 **9796-2, Digital signature scheme 2 or Digital Signature scheme 3, for RSA schemes**

- 1 • **FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 6 and Appendix D,**
2 **Implementing “NIST curves” [selection: P-256, P-384, P-521]; ISO/IEC 14888-3,**
3 **Section 6.4, for ECDSA schemes**

4].

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

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

11 **FCS_COP.1.1(b) Refinement:** The TSF shall perform [cryptographic hashing services] in
12 accordance with a specified cryptographic algorithm [**selection: SHA-256, SHA-384, SHA-**
13 **512**]-and cryptographic key sizes [assignment: cryptographic key sizes] that meet the
14 following: [ISO/IEC 10118-3:2004].

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

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

21 **FCS_COP.1.1(c) Refinement:** The TSF shall perform cryptographic [keyed-hash message
22 authentication] in accordance with a specified cryptographic algorithm [**selection: HMAC-**
23 **SHA-256, HMAC-SHA-384, HMAC-SHA-512**] and cryptographic key sizes [assignment:
24 key size (in bits) used in HMAC] that meet the following: [ISO/IEC 9797-2:2011, Section 7
25 “MAC Algorithm 2”].

26 **Application Note:** The key size [k] in the assignment falls into a range between L1 and L2
27 (defined in ISO/IEC 10118 for the appropriate hash function for example for SHA-256 L1 =
28 512, L2 =256) where $L2 \leq k \leq L1$.

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

30 **FCS_COP.1.1(d) Refinement:** The TSF shall perform [key wrapping] in accordance with a
31 specified cryptographic algorithm [AES] in the following modes [**selection: KW, KWP,**
32 **GCM, CCM**] and the cryptographic key size [**selection: 128 bits, 256 bits**] that meet the
33 following: [AES as specified in ISO/IEC 18033-3, [**selection: NIST SP 800-38F, ISO/IEC**
34 **19772, no other standards**]].

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

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

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

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

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

11 **FCS_COP.1.1(f) Refinement:** The TSF shall perform [data encryption and decryption] in
12 accordance with a specified cryptographic algorithm [AES used in [selection: CBC, GCM,
13 XTS mode]] and cryptographic key sizes [selection: 128 bits, 256 bits] that meet the
14 following: [AES as specified in ISO /IEC 18033-3, [selection: CBC as specified in ISO/IEC
15 10116, GCM as specified in ISO/IEC 19772, XTS as specified in IEEE 1619]].

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

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

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

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

34 **Application Note:** This requirement is used in the body of the ST if the ST author chooses to
35 use AES encryption/decryption for protecting the keys as part of the key chaining approach
36 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-384, SHA-512], with
20 [assignment: positive integer of 1000 or more] iterations, and output cryptographic key sizes
21 [selection: 128 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 process*
26 *described in NIST SP 800-132; the method used is selected by the ST author. If 800-132*
27 *conditioning is specified, then the ST author fills in the number of iterations that are performed.*
28 *800-132 also requires the use of a pseudo-random function (PRF) consisting of HMAC with*
29 *an approved hash function. The ST author selects the hash function used which also includes*
30 *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)]
-

1 with a minimum of [selection: 128 bits, 256 bits] of entropy at least equal to the greatest
2 security strength, according to ISO/IEC 18031:2011 Table C.1 “Security Strength Table for
3 Hash Functions”, of the keys and hashes that it will generate.

4 **Application Note:** ISO/IEC 18031:2011 contains different methods of generating random
5 numbers; each of these, in turn, depends on underlying cryptographic primitives (hash
6 functions/ciphers). The ST author will select the function used and include the specific
7 underlying cryptographic primitives used in the requirement. While any of the identified hash
8 functions (SHA-256, SHA-384, SHA-512) are allowed for Hash_DRBG or HMAC_DRBG,
9 only AES-based implementations for CTR_DRBG are allowed. Table C.2 in ISO/IEC
10 18031:2011 provides an identification of Security strengths, Entropy and Seed length
11 requirements for the AES-128 and 256 Block Cipher.

12 The CTR_DRBG in ISO/IEC 18031:2011 requires using derivation function, whereas NIST
13 SP 800-90A does not. Either model is acceptable. In the first selection in FCS_RBG_EXT.1.1,
14 the ST author chooses the standard to which the TSF is compliant.

15 In the first selection in FCS_RBG_EXT.1.2 the ST author fills in how many entropy sources
16 are used for each type of entropy source they employ. It should be noted that a combination
17 of hardware and software based noise sources is acceptable.

18 It should be noted that the entropy source is considered to be a part of the DRBG and if the
19 DRBG is included in the TOE, the developer is required to provide the entropy description
20 outlined in Appendix D. The documentation *and tests* required in the Evaluation Activity for
21 this element necessarily cover each source indicated in FCS_RBG_EXT.1.2. Individual
22 contributions to the entropy pool may be combined to provide the minimum amount of entropy
23 as long as the Entropy Documentation demonstrates that entropy from each of these individual
24 sources is generated independently.

25 **FCS_SMC_EXT.1 Submask Combining**

26
27 **FCS_SMC_EXT.1.1** The TSF shall combine submasks using the following method
28 [selection: exclusive OR (XOR), SHA-256, SHA-384, SHA-512] to generate an
29 [intermediary key or BEV].

30 **Application Note:** This requirement specifies the way that a product may combine the
31 various submasks by using either an XOR or an approved SHA-hash. The approved hash
32 functions are captured in FCS_COP.1(b).

33 **FCS_VAL_EXT.1 Validation**

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

- 36
- key wrap as specified in FCS_COP.1(d);

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

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

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

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

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

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

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

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

31 *The ST author may need to iterate this requirement if multiple authentication factors are*
32 *used, and either different methods are used to validate, or in some cases one or more*
33 *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 for
6 these dependencies may differ if the same extended components are used in other Protection
7 Profiles.

8 C.1 Background and Scope

9 This document provides a definition for all of the extended components used in this cPP. These
10 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_PWR_EXT Power Management
	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 that
13 are defined in this cPP. This definition mandates that these dependencies be included in a PP
14 that claims the SFR but it does not mandate that the dependent SFRs are defined using the same
15 iteration identifiers (e.g. inclusion of FCS_KDF_EXT.1 does not require the dependent SFR
16 for keyed-hash message authentication to be identified specifically as FCS_COP.1(c), only that
17 an FCS_COP.1 iteration exists and defines the same behavior as what this cPP defines as
18 FCS_COP.1(c)).

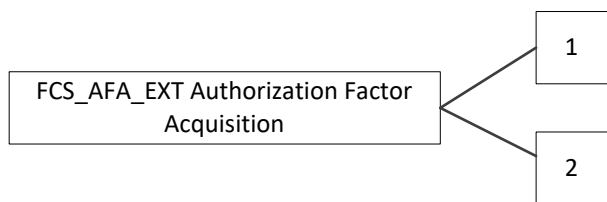
19 C.2 Extended Component Definitions

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
 - Generate external authorization factors using the TSF DRBG
- 10

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

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.

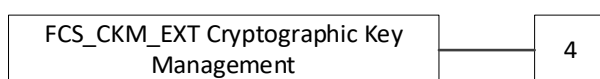
18 ***FCS_CKM_EXT Cryptographic Key Management***

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**



30 FCS_CKM_EXT.4, Key and Key Material Destruction, requires the TSF to specify
31 circumstances when keys are destroyed (as opposed to the actual method of destruction, which

1 is defined in CC Part 2 as FCS_CKM.4). The number 4 was chosen to reflect the similarity
2 between the two SFRs.

3 **Management: FCS_CKM_EXT.4**

4 No specific management functions are identified.

5 **Audit: FCS_CKM_EXT.4**

6 There are no auditable events foreseen.

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

12 Hierarchical to: No other components

13 Dependencies: No dependencies

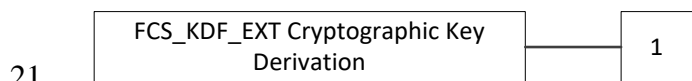
14 **FCS_CKM_EXT.4.1** The TSF shall destroy all keys and key material when no longer
15 needed.

16 ***FCS_KDF_EXT Cryptographic Key Derivation***

17 **Family Behavior**

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

20 **Component Leveling**



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

24 **Management: FCS_KDF_EXT.1**

25 No specific management functions are identified.

1 **Audit: FCS_KDF_EXT.1**

2 There are no auditable events foreseen.

3 ***FCS_KDF_EXT.1 Cryptographic Key Derivation***

4 Hierarchical to: No other components

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

6 **FCS_KDF_EXT.1.1** The TSF shall accept [selection: a RNG generated submask as specified
7 in FCS_RBG_EXT.1, a conditioned password submask, imported submask] to derive an
8 intermediate key, as defined in [selection:

- 9 • NIST SP 800-108 [selection: KDF in Counter Mode, KDF in Feedback Mode, KDF
10 in Double-Pipeline Iteration Mode],
11 • NIST SP 800-132],

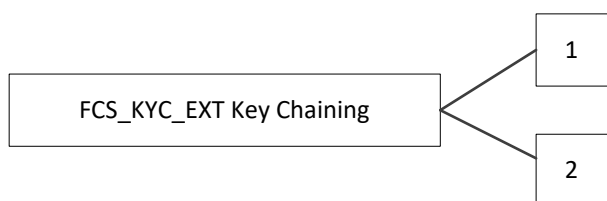
12 using the keyed-hash functions specified in FCS_COP.1(c), such that the output is at least of
13 equivalent security strength (in number of bits) to the BEV.

14 ***FCS_KYC_EXT Key Chaining***

15 **Family Behavior**

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

18 **Component Leveling**



19

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

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

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

26 **Management: FCS_KYC_EXT.1**

27 No specific management functions are identified.

1 **Audit: FCS_KYC_EXT.1**

2 There are no auditable events foreseen.

3 **Management: FCS_KYC_EXT.2**

4 No specific management functions are identified.

5 **Audit: FCS_KYC_EXT.2**

6 There are no auditable events foreseen.

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

8 Hierarchical to: No other components

9 Dependencies: FCS_CKM.1(a) Cryptographic Key Generation (Asymmetric Keys),
10 FCS_CKM.1(b) Cryptographic Operation (Symmetric Keys),
11 FCS_COP.1(d) Cryptographic Operation (Key Wrapping),
12 FCS_COP.1(e) Cryptographic Operation (Key Transport),
13 FCS_COP.1(g) Cryptographic Operation (Key Encryption),
14 FCS_SMC_EXT.1 Submask Combining,
15 FCS_VAL_EXT.1 Validation

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

- 17 • one, using a submask as the BEV;
18 • intermediate keys generated by the TSF using the following method(s): [selection:
19 o asymmetric key generation as specified in FCS_CKM.1(a),
20 o symmetric key generation as specified in FCS_CKM.1(b)];
21 • intermediate keys originating from one or more submask(s) to the BEV using the
22 following method(s): [selection:
23 o key derivation as specified in FCS_KDF_EXT.1,
24 o key wrapping as specified in FCS_COP.1(d),
25 o key combining as specified in FCS_SMC_EXT.1,
26 o key transport as specified in FCS_COP.1(e),
27 o key encryption as specified in FCS_COP.1(g)]]

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

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

- 33 • after the TSF has successfully performed the validation process as specified in
34 FCS_VAL_EXT.1,

- 1 • without validation taking place].

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

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

8 Hierarchical to: No other components

9 Dependencies: No dependencies

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

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

- 14 • asymmetric key generation as specified in FCS_CKM.1(a)
15 • symmetric key generation as specified in FCS_CKM.1(b)
16 • key derivation as specified in FCS_KDF_EXT.1,
17 • key wrapping as specified in FCS_COP.1(d),
18 • key transport as specified in FCS_COP.1(e),
19 • key encryption as specified in FCS_COP.1(g)]

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

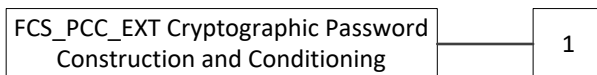
24 **Application Note:** *Key Chaining is the method of using multiple layers of encryption keys to*
25 *ultimately secure the protected data encrypted on the drive. The number of intermediate keys*
26 *will vary – from one (e.g., using the BEV as a key encrypting key (KEK)) to many. This*
27 *applies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

29 **FCS_PCC_EXT Cryptographic Password Construction and Conditioning**

30 **Family Behavior**

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

33 **Component Leveling**



1

2 FCS_PCC_EXT.1, Cryptographic Password Construction and Conditioning, requires the TSF
3 to accept passwords of a certain composition and condition them appropriately.

4 **Management: FCS_PCC_EXT.1**

5 No specific management functions are identified.

6 **Audit: FCS_PCC_EXT.1**

7 There are no auditable events foreseen.

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

9 Hierarchical to: No other components

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

11 **FCS_PCC_EXT.1.1** A password used by the TSF to generate a password authorization factor
12 shall enable up to [*assignment: positive integer of 64 or more*] characters in the set of {upper case
13 characters, lower case characters, numbers, and [*assignment: other supported special*
14 *characters*]} and shall perform Password-based Key Derivation Functions in accordance with a
15 specified cryptographic algorithm HMAC-[*selection: SHA-256, SHA-384, SHA-512*], with
16 [*assignment: positive integer of 1000 or more*] iterations, and output cryptographic key sizes
17 [*selection: 128 bits, 256 bits*] that meet the following: [*assignment: PBKDF recommendation or*
18 *specification*].

19 ***FCS_RBG_EXT Random Bit Generation***

20 **Family Behavior**

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

23 **Component Leveling**



24

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

27 **Management: FCS_RBG_EXT.1**

28 No specific management functions are identified.

1 **Audit: FCS_RBG_EXT.1**

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

- 4
 - Failure of the randomization process

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

6 Hierarchical to: No other components

7 Dependencies: FCS_COP.1(b) Cryptographic Operation (Hash Algorithm),
8 FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm)

9 **FCS_RBG_EXT.1.1** The TSF shall perform all deterministic random bit generation services
10 in accordance with [selection: ISO/IEC 18031:2011, [assignment: other RBG standards]] using
11 [selection: Hash_DRBG (any), HMAC_DRBG (any), CTR_DRBG (AES)].

12 **FCS_RBG_EXT.1.2** The deterministic RBG shall be seeded by at least one entropy source
13 that accumulates entropy from [selection:

- 14
 - [assignment: number of software-based sources] software-based noise source(s),
 - [assignment: number of hardware-based sources] hardware-based noise source(s)]

15 with a minimum of [selection: 128 bits, 256 bits] of entropy at least equal to the greatest
16 security strength, according to ISO/IEC 18031:2011 Table C.1 “Security Strength Table for
17 Hash Functions”, of the keys and hashes that it will generate.
18

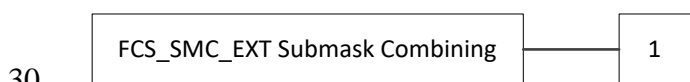
19 *Application Note: ISO/IEC 18031:2011 contains three different methods of generating*
20 *random numbers; each of these, in turn, depends on underlying cryptographic primitives*
21 *(hash functions/ciphers). The ST author will select the function used, and include the specific*
22 *underlying cryptographic primitives used in the requirement. While any of the identified hash*
23 *functions (SHA-256, SHA-384, SHA-512) are allowed for Hash_DRBG or HMAC_DRBG,*
24 *only AES-based implementations for CTR_DRBG are allowed.*

25 **FCS_SMC_EXT Submask Combining**

26 **Family Behavior**

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

29 **Component Leveling**



1 FCS_SMC_EXT.1, Submask Combining, requires the TSF to combine the submasks in a
2 predictable fashion.

3 **Management: FCS_SMC_EXT.1**

4 No specific management functions are identified.

5 **Audit: FCS_SMC_EXT.1**

6 There are no auditable events foreseen.

7 **FCS_SMC_EXT.1 Submask Combining**

8 Hierarchical to: No other components

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

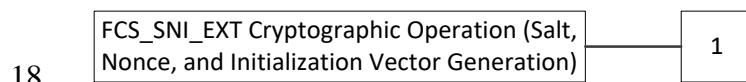
10 **FCS_SMC_EXT.1.1** The TSF shall combine submasks using the following method
11 [selection: exclusive OR (XOR), SHA-256, SHA-384, SHA-512] to generate an [*assignment:*
12 *types of keys*].

13 ***FCS_SNI_EXT Cryptographic Operation (Salt, Nonce, and Initialization Vector***
14 ***Generation)***

15 **Family Behavior**

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

17 **Component Leveling**



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

22 **Management: FCS_SNI_EXT.1**

23 No specific management functions are identified.

24 **Audit: FCS_SNI_EXT.1**

25 There are no auditable events foreseen.

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

1 Hierarchical to: No other components

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

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

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

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

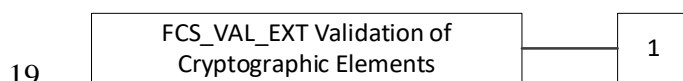
- 8 • CBC: IVs shall be non-repeating and unpredictable;
- 9 • CCM: Nonce shall be non-repeating and unpredictable;
- 10 • XTS: No IV. Tweak values shall be non-negative integers, assigned consecutively,
11 and starting at an arbitrary non-negative integer;
- 12 • GCM: IV shall be non-repeating. The number of invocations of GCM shall not exceed
13 2³² for a given secret key].

14 FCS_VAL_EXT Validation of Cryptographic Elements

15 Family Behavior

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

18 Component Leveling



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

22 Management: FCS_VAL_EXT.1

23 No specific management functions are identified.

24 Audit: FCS_VAL_EXT.1

25 There are no auditable events foreseen.

26 FCS_VAL_EXT.1 Validation

27 Hierarchical to: No other components

1 Dependencies: FCS_COP.1(b) Cryptographic Operation (Hash Algorithm),
2 FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm),
3 FCS_COP.1(d) Cryptographic Operation (Key Wrapping),
4 FCS_COP.1(f) Cryptographic Operation (AES Data
5 Encryption/Decryption)

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

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

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

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

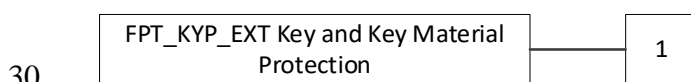
- 17 • perform a key sanitization of the DEK upon a [selection: configurable number,
18 [assignment: *ST author specified number*]] of consecutive failed validation attempts,
- 19 • institute a delay such that only [assignment: *ST author specified number of attempts*]
20 can be made within a 24 hour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

25 ***FPT_KYP_EXT Key and Key Material Protection***

26 **Family Behavior**

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

29 **Component Leveling**



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

1 **Management: FPT_KYP_EXT.1**

2 No specific management functions are identified.

3 **Audit: FPT_KYP_EXT.1**

4 There are no auditable events foreseen.

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

6 Hierarchical to: No other components

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

13 **FPT_KYP_EXT.1.1** The TSF shall [selection:

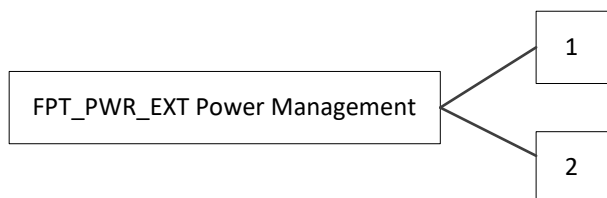
- 14 • not store keys in non-volatile memory
- 15 • only store keys in non-volatile memory when wrapped, as specified in
16 FCS_COP.1(d), or encrypted, as specified in FCS_COP.1(g) or FCS_COP.1(e)
- 17 • only store plaintext keys that meet any one of the following criteria [selection:
 - 18 ○ the plaintext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

35 ***FPT_PWR_EXT Power Management***

36 **Family Behavior**

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

5 **Component Leveling**



6

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

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

11 **Management: FPT_PWR_EXT.1**

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

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

15 **Audit: FPT_PWR_EXT.1**

16 There are no auditable events foreseen.

17 **Management: FPT_PWR_EXT.2**

18 There are no management activities foreseen.

19 **Audit: FPT_PWR_EXT.2**

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

- 22 • Transition of the TSF into different power saving states

23 **FPT_PWR_EXT.1 Authorization Factor Acquisition**

24 Hierarchical to: No other components

25 Dependencies: No dependencies

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

3 **FPT_PWR_EXT.2 Authorization Factor Acquisition**

4 Hierarchical to: No other components

5 Dependencies: FPT_PWR_EXT.1 Power Saving States

6 **FPT_PWR_EXT.2.1** For each Compliant power saving state defined in
7 FPT_PWR_EXT.1.1, the TSF shall enter the Compliant power saving state when the
8 following conditions occur: user-initiated request, [selection: shutdown, user inactivity,
9 request initiated by remote management system, [assignment: other conditions], no other
10 conditions].

11 ***FPT_TST_EXT TSF Testing***

12 **Family Behavior**

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

15 **Component Leveling**



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

19 **Management: FPT_TST_EXT.1**

20 No specific management functions are identified.

21 **Audit: FPT_TST_EXT.1**

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

- 24
- Indication that TSF self-test was completed

25 **FPT_TST_EXT.1 TSF Testing**

26 Hierarchical to: No other components

27 Dependencies: No other components

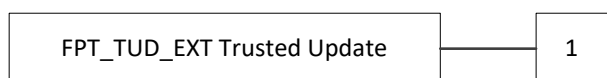
1 **FPT_TST_EXT.1.1** The TSF shall run a suite of the following self-tests [selection: during
2 initial start-up (on power on), periodically during normal operation, at the request of the
3 authorized user, at the conditions [assignment: conditions under which self-tests should
4 occur] to demonstrate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

6 **FPT_TUD_EXT Trusted Update**

7 **Family Behavior**

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

10 **Component Leveling**



11
12 FPT_TUD_EXT.1, Trusted Update, requires the capability to be provided to update the TOE
13 firmware and software, including the ability to verify the updates prior to installation.

14 **Management: FPT_TUD_EXT.1**

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

- 16 • Ability to update the TOE and to verify the updates

17 **Audit: FPT_TUD_EXT.1**

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

- 20 • Initiation of the update process.
- 21 • Any failure to verify the integrity of the update

22 **FPT_TUD_EXT.1 Trusted Update**

23 Hierarchical to: No other components

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

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

- 1 **FPT_TUD_EXT.1.2** The TSF shall provide [*assignment: list of subjects*] the ability to initiate
2 updates to TOE software/firmware.
- 3 **FPT_TUD_EXT.1.3** The TSF shall verify updates to the TOE software/firmware using a
4 [selection: digital signature, published hash] by the manufacturer prior to installing those
5 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 deterministic*
3 *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 the
14 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 is
17 produced, and how unprocessed (raw) data can be obtained from within the entropy source for
18 testing purposes. The documentation should walk through the entropy source design indicating
19 where the entropy comes from, where the entropy output is passed next, any post-processing
20 of the raw outputs (hash, XOR, etc.), if/where it is stored, and finally, how it is output from the
21 entropy source. Any conditions placed on the process (e.g., blocking) should also be described
22 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 outside
25 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 data)
34 and explain that sufficient entropy is going into the TOE randomizer seeding process. This
35 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, and
6 the min-entropy rate determined from the statistical tests. While no particular statistical tests
7 are required at this time, it is expected that some testing is necessary in order to determine the
8 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 to
13 “assume” an amount of min-entropy, however, this assumption must be clearly stated in the
14 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 rate
18 is multiplied by the amount of source data used to seed the DRBG or that the rate of entropy
19 expected based on the amount of source data is explicitly stated and compared to the statistical
20 rate. If the amount of source data used to seed the DRBG is not clear or the calculated rate is
21 not explicitly related to the seed, the documentation will not be considered complete.

22
23 The entropy justification shall not include any data added from any third-party application or
24 from any state saving between restarts.

25 **D.3 Operating Conditions**

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

33 **D.4 Health Testing**

34 More specifically, all entropy source health tests and their rationale will be documented. This
35 will include a description of the health tests, the rate and conditions under which each health
36 test is performed (e.g., at startup, continuously, or on-demand), the expected results for each
37 health test, TOE behavior upon entropy source failure, and rationale indicating why each test
38 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 and
4 how it meets the requirements to ensure the keys are adequately protected. This documentation
5 should include an essay and diagram(s). This documentation is not required to be part of the
6 TSS - it can be submitted as a separate document and marked as developer proprietary.

7 Essay:

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

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

15 The essay will also describe the following topics:

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

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

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

11 Diagram:

- 12 • The diagram will include all keys from the initial authorization factor(s) to the BEV
13 and any keys or values that contribute into the chain. It must list the cryptographic
14 strength of each key and indicate how each key along the chain is protected with
15 either Key Derivation or Key Wrapping (from the allowed options). The diagram
16 should indicate the input used to derive or unwrap each key in the chain.
- 17 • A functional (block) diagram showing the main components (such as memories and
18 processors) and the data path between, for hardware, the device's host interface and
19 the device's persistent media storing the data, or for software, the initial steps needed
20 for the activities the TOE performs to ensure it encrypts the storage device entirely
21 when a user or administrator first provisions the product. The hardware encryption
22 diagram shall show the location of the data encryption engine within the data path.
- 23 • The hardware encryption diagram shall show the location of the data encryption
24 engine within the data path. The evaluator shall validate that the hardware encryption
25 diagram contains enough detail showing the main components within the data path
26 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.
Operating System (OS)	Software which runs at the highest privilege level and can directly control hardware resources.

Term	Meaning
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

1 **Appendix H: References**

- 2 National Institute of Standards and Technology (NIST) Special Publication 800-38F,
3 Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping, National
4 Institute of Standards and Technology, December 2012.
- 5 National Institute of Standards and Technology (NIST) Special Publication 800-56B,
6 Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization
7 Cryptography, National Institute of Standards and Technology, August 2009.
- 8 National Institute of Standards and Technology (NIST) Special Publication 800-88 Revision
9 1, Guidelines for Media Sanitization, National Institute of Standards and Technology,
10 December 2014.
- 11 National Institute of Standards and Technology (NIST) Special Publication 800-90A,
12 Recommendation for Random Number Generation Using Deterministic Random Bit
13 Generators, National Institute of Standards and Technology, January 2012.
- 14 National Institute of Standards and Technology (NIST) Special Publication 800-132,
15 Recommendation for Password-Based Key Derivation Part 1: Storage Applications, National
16 Institute of Standards and Technology, December 2010.
- 17 Federal Information Processing Standard Publication (FIPS-PUB) 186-4, Digital Signature
18 Standard (DSS), National Institute of Standards and Technology, July 2013.
- 19 International Organization for Standardization (ISO)/International Electrotechnical
20 Commission (IEC) 9796-2:2010 (3rd edition), Information technology — Security techniques
21 — Digital signature schemes giving message recovery, International Organization for
22 Standardization/International Electrotechnical Commission, 2010.
- 23 International Organization for Standardization (ISO)/International Electrotechnical
24 Commission (IEC) 9797-2:2011 (2nd edition), Information technology — Security techniques
25 — Message Authentication Codes (MACs) – Part 2: Mechanisms using a dedicated hash-
26 function, International Organization for Standardization/International Electrotechnical
27 Commission, 2011.
- 28 International Organization for Standardization (ISO)/International Electrotechnical
29 Commission (IEC) 10116:2006 (3rd edition), Information technology — Security techniques
30 — Modes of operation for an n-bit block cipher, International Organization for
31 Standardization/International Electrotechnical Commission, 2006.
- 32 International Organization for Standardization (ISO)/International Electrotechnical
33 Commission (IEC) 10118-3:2004 (3rd edition), Information technology — Security techniques
34 — Hash-functions – Part 3: Dedicated hash-functions, International Organization for
35 Standardization/International Electrotechnical Commission, 2004.

- 1 International Organization for Standardization (ISO)/International Electrotechnical
2 Commission (IEC) 14888-3:2006 (2nd edition), Information technology — Security techniques
3 —Digital signatures with appendix – Part 3: Discrete logarithm based mechanisms,
4 International Organization for Standardization/International Electrotechnical Commission,
5 2006.
- 6 International Organization for Standardization (ISO)/International Electrotechnical
7 Commission (IEC) 18031:2011 (2nd edition), Information technology — Security techniques
8 — Random bit generation, International Organization for Standardization/International
9 Electrotechnical Commission, 2011.
- 10 International Organization for Standardization (ISO)/International Electrotechnical
11 Commission (IEC) 18033-3:2011 (3rd edition), Information technology — Security techniques
12 — Encryption algorithms – Part 3: Block ciphers, International Organization for
13 Standardization/International Electrotechnical Commission, 2011.
- 14 International Organization for Standardization (ISO)/International Electrotechnical
15 Commission (IEC) 19772:2009, Information technology — Security techniques Authenticated
16 encryption, International Organization for Standardization/International Electrotechnical
17 Commission, 2009.