collaborative Protection Profile for Dedicated Security Component

Version 1.0d

May 1, 2019

Acknowledgements

This collaborative Protection Profile (cPP) was developed by the Dedicated Security Components international Technical Community with representatives from Industry, Common Criteria Test Laboratories, and International Common Criteria schemes. The organizations that directly contributed to the development of this cPP include:

**Industry**

Apple Inc.

Samsung Electronics Co., Ltd.

**Common Criteria Test Laboratories**

atsec information security

Det Norske Veritas and Germanischer Lloyd (DNV GL)

**International Common Criteria Schemes**

National Information Assurance Partnership (NIAP)

UK IT Security Evaluation and Certificate Scheme (NCSC)

# Preface

## Objectives of Document

This document presents the Common Criteria (CC) collaborative Protection Profile (cPP) to express the Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs) for a Dedicated Security Component (DSC). The Evaluation Activities that specify the actions an evaluator performs to determine if a product satisfies the SFRs/SARs captured within this cPP are described in the Evaluation Activities for Dedicated Security Component cPP Supporting Document [SD].

The DSC international Technical Community (iTC) designed the DSC cPP to stand alone so that vendors may evaluate their DSCs alone against it. Vendors may also combine this cPP with a platform solution cPP for CC consumers.

## Scope of Document

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

## Intended Readership

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

## Related Documents

**Common Criteria**[[1]](#footnote-2)

|  |  |
| --- | --- |
| [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. |

**Other Documents**

|  |  |
| --- | --- |
| [SD] | Evaluation Activities for Dedicated Security Component cPP, Version 1.0d, April 2019 |

## Revision History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Description** |
| 1.0d | 01-MAY-2019 | [DRAFT] Initial draft |

Contents

[Acknowledgements 2](#_Toc7020370)

[0 Preface 3](#_Toc7020371)

[0.1 Objectives of Document 3](#_Toc7020372)

[0.2 Scope of Document 3](#_Toc7020373)

[0.3 Intended Readership 3](#_Toc7020374)

[0.4 Related Documents 3](#_Toc7020375)

[0.5 Revision History 4](#_Toc7020376)

[1 PP Introduction 11](#_Toc7020377)

[1.1 PP Reference Identification 11](#_Toc7020378)

[1.2 TOE Overview 11](#_Toc7020379)

[1.2.1 Security Data Objects 12](#_Toc7020380)

[1.2.2 Services 14](#_Toc7020381)

[1.2.3 Roots of Trust 15](#_Toc7020382)

[1.2.4 DSC Characteristics 16](#_Toc7020383)

[1.3 TOE Use Cases 18](#_Toc7020384)

[1.4 Key Reference Model 19](#_Toc7020385)

[1.4.1 Roles 19](#_Toc7020386)

[1.4.2 Key Usage 20](#_Toc7020387)

[1.4.3 Sessions 20](#_Toc7020388)

[1.4.4 Key Hierarchies 21](#_Toc7020389)

[1.4.5 Protected Storage Locations 24](#_Toc7020390)

[1.4.6 SDEs and SDOs 24](#_Toc7020391)

[2 CC Conformance Claims 25](#_Toc7020392)

[3 Security Problem Definition 26](#_Toc7020393)

[3.1 Assets 26](#_Toc7020394)

[3.2 Threats 26](#_Toc7020395)

[3.3 Assumptions 29](#_Toc7020396)

[3.4 Organizational Security Policies 30](#_Toc7020397)

[4 Security Objectives 31](#_Toc7020398)

[4.1 Security Objectives for the TOE 31](#_Toc7020399)

[4.2 Security Objectives for the Operational Environment 37](#_Toc7020400)

[4.3 Security Objectives Rationale 38](#_Toc7020401)

[5 Security Functional Requirements 40](#_Toc7020402)

[5.1 SFR Architecture 40](#_Toc7020403)

[5.2 Conventions 44](#_Toc7020404)

[5.3 Cryptographic Support 45](#_Toc7020405)

[5.3.1 FCS\_CKM.1 Cryptographic Key Generation 45](#_Toc7020406)

[5.3.2 FCS\_CKM.1/AK Cryptographic Key Generation (Asymmetric Keys) 46](#_Toc7020407)

[5.3.3 FCS\_CKM.1/KEK Cryptographic Key Generation (Key Encryption Key) 46](#_Toc7020408)

[5.3.4 FCS\_CKM.2 Cryptographic Key Establishment 47](#_Toc7020409)

[5.3.5 FCS\_CKM.4 Cryptographic Key Destruction 47](#_Toc7020410)

[5.3.6 FCS\_CKM\_EXT.4 Cryptographic Key and Key Material Destruction Timing 49](#_Toc7020411)

[5.3.7 FCS\_CKM\_EXT.5 Cryptographic Key Derivation 50](#_Toc7020412)

[5.3.8 FCS\_COP.1/Hash Cryptographic Operation (Hashing) 52](#_Toc7020413)

[5.3.9 FCS\_COP.1/HMAC Cryptographic Operation (Keyed Hash) 53](#_Toc7020414)

[5.3.10 FCS\_COP.1/KAT Cryptographic Operation (Key Agreement/Transport) 53](#_Toc7020415)

[5.3.11 FCS\_COP.1/KeyEnc Cryptographic Operation (Key Encryption) 54](#_Toc7020416)

[5.3.12 FCS\_COP.1/PBKDF Cryptographic Operation (Password-Based Key Derivation Functions) 55](#_Toc7020417)

[5.3.13 FCS\_COP.1/SigGen Cryptographic Operation (Signature Generation) 55](#_Toc7020418)

[5.3.14 FCS\_COP.1/SigVer Cryptographic Operation (Signature Verification) 56](#_Toc7020419)

[5.3.15 FCS\_COP.1/SKC Cryptographic Operation (Symmetric Key Cryptography) 57](#_Toc7020420)

[5.3.16 FCS\_RBG\_EXT.1 Random Bit Generation 59](#_Toc7020421)

[5.3.17 FCS\_SLT\_EXT.1 Cryptographic Salt Generation 60](#_Toc7020422)

[5.3.18 FCS\_STG\_EXT.1 Secure Key Storage 60](#_Toc7020423)

[5.3.19 FCS\_STG\_EXT.2 Key Storage Encryption 61](#_Toc7020424)

[5.3.20 FCS\_STG\_EXT.3 Key Integrity Protection 61](#_Toc7020425)

[5.4 User Data Protection 62](#_Toc7020426)

[5.4.1 FDP\_ACC.1 Subset Access Control 62](#_Toc7020427)

[5.4.2 FDP\_ACF.1 Security Attribute Based Access Control 62](#_Toc7020428)

[5.4.3 FDP\_APW\_EXT.1 Storage of Authentication Tokens 64](#_Toc7020429)

[5.4.4 FDP\_ETC\_EXT.1 Secure Propagation of User Data 64](#_Toc7020430)

[5.4.5 FDP\_FRS\_EXT.1 Factory Reset 64](#_Toc7020431)

[5.4.6 FDP\_IFC.1 Subset Information Flow Control 65](#_Toc7020432)

[5.4.7 FDP\_IFF.1 Simple Security Attributes 65](#_Toc7020433)

[5.4.8 FDP\_ITC.1 Import of User Data without Security Attributes 66](#_Toc7020434)

[5.4.9 FDP\_ITC\_EXT.1 Parsing of SDEs 66](#_Toc7020435)

[5.4.10 FDP\_ITC.2 Import of User Data with Security Attributes 67](#_Toc7020436)

[5.4.11 FDP\_ITC\_EXT.2 Parsing of SDOs 67](#_Toc7020437)

[5.4.12 FDP\_MFW\_EXT.1 Mutable/Immutable Firmware 68](#_Toc7020438)

[5.4.13 FDP\_RIP.1 Subset Residual Information Protection 68](#_Toc7020439)

[5.4.14 FDP\_SDC\_EXT.1 Confidentiality of SDEs 68](#_Toc7020440)

[5.4.15 FDP\_SDI.2 Stored Data Integrity Monitoring and Action 69](#_Toc7020441)

[5.5 Identification and Authentication 69](#_Toc7020442)

[5.5.1 FIA\_AFL\_EXT.1 Authorization Failure Handling 70](#_Toc7020443)

[5.5.2 FIA\_SOS.2 TSF Generation of Secrets 70](#_Toc7020444)

[5.5.3 FIA\_UIA\_EXT.1 User Identification and Authentication before Any Action 71](#_Toc7020445)

[5.5.4 FIA\_UAU.6/SDO Re-Authenticating (Access to SDO) 72](#_Toc7020446)

[5.5.5 FIA\_USB\_EXT.1 Response to User Subject Binding 72](#_Toc7020447)

[5.6 Security Management 73](#_Toc7020448)

[5.6.1 FMT\_MOF\_EXT.1 Management of Security Functions Behavior 73](#_Toc7020449)

[5.6.2 FMT\_MSA.1/SDO Management of Security Attributes (Secure Data Objects) 73](#_Toc7020450)

[5.6.3 FMT\_MSA.3/SDO Static Attribute Initialization (Secure Data Objects) 74](#_Toc7020451)

[5.6.4 FMT\_SMF.1 Specification of Management Functions 77](#_Toc7020452)

[5.6.5 FMT\_SMR.2 Separation of Roles 77](#_Toc7020453)

[5.7 Protection of the TSF 78](#_Toc7020454)

[5.7.1 FPT\_APW\_EXT.1 Protection of Administrator Passwords 78](#_Toc7020455)

[5.7.2 FPT\_FLS.1/FI Failure with Preservation of Secure State (Fault Injection) 78](#_Toc7020456)

[5.7.3 FPT\_MOD\_EXT.1 Debug Modes 78](#_Toc7020457)

[5.7.4 FPT\_PHP.3 Resistance to Physical Attack 78](#_Toc7020458)

[5.7.5 FPT\_PRO\_EXT.1 Root of Trust 79](#_Toc7020459)

[5.7.6 FPT\_ROT\_EXT.1 Root of Trust Services 79](#_Toc7020460)

[5.7.7 FPT\_ROT\_EXT.2 Root of Trust for Storage 80](#_Toc7020461)

[5.7.8 FPT\_RPL\_EXT.1 Replay Prevention 80](#_Toc7020462)

[5.7.9 FPT\_STM.1 Reliable Time Stamps 80](#_Toc7020463)

[5.7.10 FPT\_TST.1 Integrity Checking 81](#_Toc7020464)

[5.8 Resource Utilization 81](#_Toc7020465)

[5.8.1 FRU\_FLT.1 Degraded Fault Tolerance 81](#_Toc7020466)

[6 Security Assurance Requirements 82](#_Toc7020467)

[6.1 ASE: Security Target 83](#_Toc7020468)

[6.2 ADV: Development 83](#_Toc7020469)

[6.2.1 Basic Functional Specification (ADV\_FSP.1) 83](#_Toc7020470)

[6.3 AGD: Guidance Documentation 84](#_Toc7020471)

[6.3.1 Operational User Guidance (AGD\_OPE.1) 84](#_Toc7020472)

[6.3.2 Preparative Procedures (AGD\_PRE.1) 84](#_Toc7020473)

[6.4 Class ALC: Life-cycle Support 84](#_Toc7020474)

[6.4.1 Labelling of the TOE (ALC\_CMC.1) 84](#_Toc7020475)

[6.4.2 TOE CM Coverage (ALC\_CMS.1) 84](#_Toc7020476)

[6.5 Class ATE: Tests 85](#_Toc7020477)

[6.5.1 Independent Testing – Conformance (ATE\_IND.1) 85](#_Toc7020478)

[6.6 Class AVA: Vulnerability Assessment 85](#_Toc7020479)

[6.6.1 Vulnerability Survey (AVA\_VAN.1) 85](#_Toc7020480)

[Appendix A: Optional Requirements 86](#_Toc7020481)

[A.1 Cryptographic Support 86](#_Toc7020482)

[A.1.1 FCS\_ENT\_EXT.1 Entropy for External IT Entities 86](#_Toc7020483)

[A.1.2 FCS\_RBG\_EXT.2 External Seeding for Random Bit Generation 86](#_Toc7020484)

[A.2 Protection of the TSF 86](#_Toc7020485)

[A.2.1 FPT\_PRO\_EXT.2 Data Integrity Measurements 86](#_Toc7020486)

[A.2.2 FPT\_ROT\_EXT.3 Root of Trust for Reporting Mechanisms 87](#_Toc7020487)

[Appendix B: Selection-Based Requirements 89](#_Toc7020488)

[B.1 Cryptographic Support 89](#_Toc7020489)

[B.1.1 FCS\_CKM.1/SK Cryptographic Key Generation (Symmetric Encryption Key) 89](#_Toc7020490)

[B.2 User Data Protection 90](#_Toc7020491)

[B.2.1 FDP\_DAU.1/Prove Data Authentication for Use with The Prove Service 90](#_Toc7020492)

[B.2.2 FDP\_FRS\_EXT.2 Factory Reset Behavior 91](#_Toc7020493)

[B.2.3 FDP\_MFW\_EXT.2 Basic Firmware Integrity 91](#_Toc7020494)

[B.2.4 FDP\_MFW\_EXT.3 Firmware Authentication with Identity of Guarantor 92](#_Toc7020495)

[B.3 Identification and Authentication 92](#_Toc7020496)

[B.3.1 FIA\_AFL\_EXT.2 Authorization Failure Response 92](#_Toc7020497)

[B.3.2 FIA\_UAU.5/Prove Multiple Authentication Mechanisms (Prove Service) 93](#_Toc7020498)

[B.4 Protection of the TSF 93](#_Toc7020499)

[B.4.1 FPT\_FLS.1/FW Failure with Preservation of Secure State (Firmware) 93](#_Toc7020500)

[B.4.2 FPT\_ITT.1 Basic Internal TSF Data Transfer Protection 94](#_Toc7020501)

[B.4.3 FPT\_RPL.1/Rollback Replay Detection (Rollback) 94](#_Toc7020502)

[B.5 Trusted Path/Channels 94](#_Toc7020503)

[B.5.1 FTP\_CCMP\_EXT.1 CCM Protocol 94](#_Toc7020504)

[B.5.2 FTP\_GCMP\_EXT.1 GCM Protocol 95](#_Toc7020505)

[B.5.3 FTP\_ITC\_EXT.1 Cryptographically Protected Communications Channels 95](#_Toc7020506)

[B.5.4 FTP\_ITE\_EXT.1 Encrypted Data Communications 95](#_Toc7020507)

[B.5.5 FTP\_ITP\_EXT.1 Physically Protected Channel 96](#_Toc7020508)

[Appendix C: Extended Component Definitions 97](#_Toc7020509)

[Appendix D: Entropy Documentation and Assessment 98](#_Toc7020510)

[D.1 Design Description 98](#_Toc7020511)

[D.2 Entropy Justification 98](#_Toc7020512)

[D.3 Operating Conditions 99](#_Toc7020513)

[D.4 Health Testing 99](#_Toc7020514)

[Appendix E: Glossary 100](#_Toc7020515)

[Appendix F: Acronyms 104](#_Toc7020516)

[Appendix G: References 108](#_Toc7020517)

Figures and Tables

[Figure 1: Representation of the Target of Evaluation (TOE) 11](#_Toc6412707)

[Figure 2: Example of TOE Internal Components 12](#_Toc6412708)

[Figure 3: Composition of a Security Data Object 12](#_Toc6412709)

[Figure 4: Services Provided by the TOE 14](#_Toc6412710)

[Figure 5: Example Key Hierarchy 23](#_Toc6412711)

[Table 1: Core Security Services 15](#_Toc6412712)

[Table 2: Security Problem Definition Mapping to Security Objectives 39](#_Toc6412713)

[Table 3: SFR Architecture 43](#_Toc6412714)

[Table 4: Sample Cryptographic Table 45](#_Toc6412715)

[Table 5: Supported Methods for Asymmetric Key Generation 46](#_Toc6412716)

[Table 6: Key Derivation Functions 51](#_Toc6412717)

[Table 7: Supported Methods for Key Agreement/Transport Operation 54](#_Toc6412718)

[Table 8: Supported Methods for Key Encryption Operation 54](#_Toc6412719)

[Table 9: Supported Methods for Signature Generation Operation 56](#_Toc6412720)

[Table 10: Supported Methods for Signature Verification Operation 57](#_Toc6412721)

[Table 11: Supported Methods for Symmetric Key Cryptography Operation 59](#_Toc6412722)

[Table 12: Supported Methods for SDO Attributes Security Attributes 74](#_Toc6412723)

[Table 13: Supported Methods for Secure Data Objects Attributes Initialization 75](#_Toc6412724)

[Table 14: Security Assurance Requirements 83](#_Toc6412725)

[Table 15: Supported Methods for Symmetric Encryption Key Generation 89](#_Toc6412726)

[Table 16: Glossary 103](#_Toc6412727)

[Table 17: Acronyms 107](#_Toc6412728)

# PP Introduction

## PP Reference Identification

PP Reference: collaborative Protection Profile for Dedicated Security Component

PP Version: 1.0d

PP Date: May 1, 2019

## TOE Overview

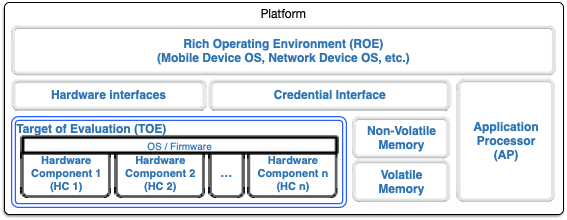
The Target of Evaluation (TOE) is a Dedicated Security Component (DSC). In the context of this cPP, a DSC is the combination of a hardware component and its controlling firmware dedicated to providing the encompassing platform with services for the provisioning, protection, and use of Security Data Objects (SDOs) consisting of keys, identities, attributes, and other types of Security Data Elements (SDEs). See Figure 1 for an example of a TOE representation.

Figure 1: Representation of the Target of Evaluation (TOE)

The TOE would be discrete and embedded hardware components providing well-scoped security functions physically inaccessible directly from the rich operating system. The DSC TOE would consist of isolated firmware and circuitry capable of executing well-defined commands against SDEs/SDOs in memory and across restricted interfaces.

Examples of a DSC that could claim conformance to this cPP include Secure Elements (SE), Trusted Platform Modules (TPM), Hardware Security Modules (HSM), Trusted Execution Environments (TEE), and Secure Enclave Processors (SEP). Already in some cases, vendors integrated these dedicated hardware components into a System on Chip (SoC) and as such are isolated components of a larger physical package. Figure 2 below shows a block diagram of a typical example of a DSC TOE with all of its internal components.

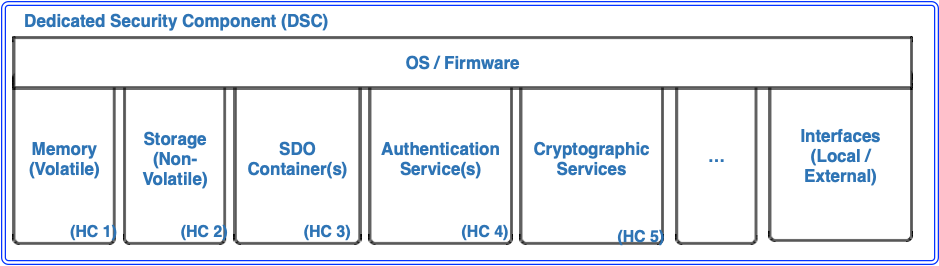
Protection profiles normally include FAU\_ or audit SFRs. However, the DSC protection profiles does not include them at this time due to the fact that DSC-style implementations lack the resources to deploy a complete subsystem for auditing and the time constraints under which the iTC worked to publish this version. The iTC expects to include FAU\_ SFRs in a future release of the DSC cPP.

Figure 2: Example of TOE Internal Components

### Security Data Objects

Figure 3: Composition of a Security Data Object

An SDO is created by combining SDEs with some attributes. Each SDE used to create the SDO reaches the DSC in either of two ways:

1. by parsing SDEs received via secure channels (see O.PARSE\_PROTECTION); or
2. by generating the SDEs locally on the DSC as part of Provisioning.

An SDO may include one or more SDEs from one or both of these sources. In the Provisioning step the relevant SDE(s) are then bound together with a set of attributes resulting in an SDO. Explicit binding occurs when the DSC includes one or more SDEs along with their attributes in a formatted structure to form the SDO. An X.509 certificate is just one example of an SDO (where the signature in the certificate provides the binding of the attributes contained). A DSC protects the integrity of an SDO with one of the following methods:

* + 1. Hash/keyed hash (FCS\_COP.1/Hash, FCS\_COP.1/HMAC)
    2. Digital signature (FCS\_COP.1/SigGen, FCS\_COP.1/SigVer)

Explicit binding may also occur when the DSC wraps an SDO prior to storing it externally. Figure 3 above shows an example SDO with binding data used to secure an arbitrary number of SDEs.

Implicit binding may occur by virtue of the location of SDEs within the DSC. An implicit binding may occur for pre-installed SDEs, in which case the DSC restricts the functionality it allows with the SDEs. It may also occur when the contents of certain protected storage locations carry with them implicit attributes simply by existing in these locations.

Sometimes vendors will pre-install keys and other material in the DSC during the manufacturing process, or the DSC will automatically generate keys or other material upon first boot. Since the user provides no input to these items, we call these pre-installed SDEs. They have two distinguishing characteristics: (a) these keys may persist over a factory reset; and (b) they may not be accessible to administrators. If the SDOs have been erased (e.g. due to a tamper response), then a factory reset may not be possible. Following an initial boot (e.g. first boot by end-user, or following a factory reset), a DSC may generate SDEs unique to an instance of a DSC and persist across user sessions, are considered pre-installed SDEs. Pre-installed SDOs (i.e., SDEs with implicit binding installed by the vendor at manufacturing time) are typically not accessible by just any non-administrative users of the platform and are reserved for use by the DSC itself to manage its sub-components, keys, and, indirectly, user content. Pre-installed SDOs typically have implicitly bound attributes. Since pre-installed SDOs rarely, if ever, leave the DSC, they may have no formal structure containing attributes. That does not mean these attributes do not exist; only that there exists no structure in which one would find them all in one place. Nonetheless, this DSC assumes that the attributes do exist even if there exists no single place to find them all. The DSC may allow the modification of attributes for pre-installed SDOs. One example would be the authorization value necessary to use the SDO. Obviously, the vendor may have a strong desire to keep the users of the DSC from changing the SDE itself, or deleting it. They could allow users (e.g. administrators) to hide the SDO, but not delete it for the sake of factory resets.

Another case of implicit binding occurs when a DSC reserves a bank of user-accessible registers with common attributes. In this case, user can be administrative only users, non-administrative users only, or both. The bank contains one or more registers, usually all of the same size. Here, again, the functionality within the firmware determines the attributes especially when the function applies only to one or more members of the bank of reserved registers. Without the benefit of a structure with explicit attributes, the DSC relies on the firmware to enforce the policies inherent to the attributes associated with a bank of registers. I.e., the DSC firmware implicitly binds the common attributes to the bank of registers.

An SDO held in the DSC may be exported (propagated) only if it is either in a wrapped form (i.e. with confidentiality and integrity of the SDO protected by a cryptographic key-based operation), or if it is transmitted over a secure channel (protecting confidentiality, integrity and optionally authenticity of the receiving endpoint).

### Services

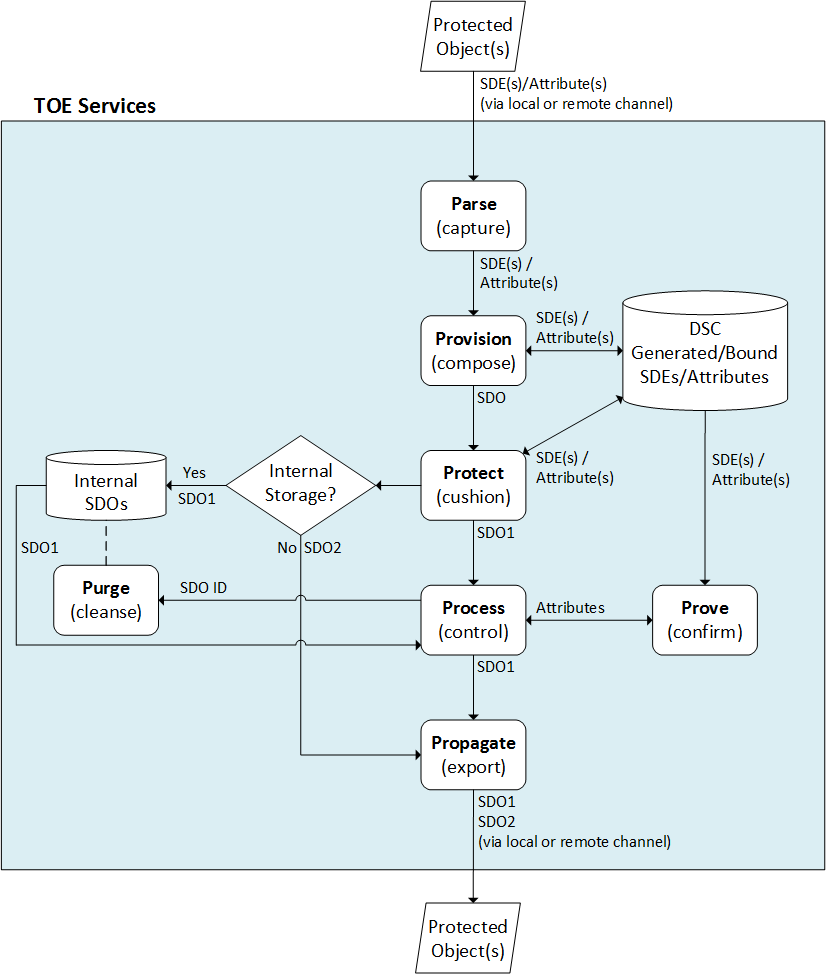


Figure 4: Services Provided by the TOE

Note the labels in Figure 4 refer to the following:

* SDE: Security Data Element
* SDO: Security Data Object (component from SDEs and attributes)
* SDO ID: Unique identifier for an SDO
* SDO1: SDO that is modified or is a reference to original SDO
* SDO2: SDO that is bound to the DSC but stored outside of it

DSCs provide seven core security services to a platform as illustrated in Table 1.

| Service | Description |
| --- | --- |
| Parse | The DSC shall ingest pre-provisioned keys, credential, tokens, attributes, etc. from trusted components or services external to its boundary either across a secured channel or in a manner that the objects are protected for use only by the DSC. |
| Provision | The DSC shall generate Security Data Objects (SDOs) from parsed or generated SDEs and attributes using binding mechanism(s) to integrity protect the SDEs together with their attributes. |
| Protect | The DSC shall manage protected storage for all SDOs. Platform users (through applications) can store SDOs either inside or outside the DSC boundary. A DSC shall maintain the integrity and confidentiality (if required) of SDOs both inside the boundary and stored outside the boundary. |
| Process | The DSC shall operate on and maintain SDOs or their attributes on behalf of authorized entities. The Process service shall coordinate with the Protect service for storage of the SDOs while not in use and shall collaborate with the Prove service to authenticate the requesting entity and shall validate their authorization for access to the SDO in the requested mode. The Process service shall submit an SDO to the Purge service when it is no longer needed by the platform. |
| Prove | The DSC may provide evidence to a remote entity that the DSC is currently in a specific state. During this process, the DSC shall use the appropriate attributes and/or authentication tokens (such as nonces, digital signatures, etc.) to enable the remote entity to verify (or "prove") the authenticity of the source of the evidence. |
| Purge | When the SDO is no longer needed by the platform, the DSC shall execute a mechanism for permanently purging the SDO from the DSC to protect against unauthorized recovery. |
| Propagate | If an SDO is required by or allowed to be used by a remote peer, the DSC shall ensure that the SDO is exported only as a protected object or is transmitted over a trusted channel. |

Table 1: Core Security Services

### Roots of Trust

It is expected that a DSC will contain one, and only one, root of trust comprised of the DSC hardware, its pre-installed Security Data Objects, and a unique identity key bound to the hardware. The root of trust provides the foundation of trust upon which the DSC cryptographically ensures the authenticity and integrity of the DSC trusted computing base, and for the authorization of both security data objects and access to those objects. Depending on the use case and the way status registers are utilized, unique identity keys may be bound to either the TOE, TOE platform or both.

It is expected that the root of trust identity contained within the Security Data Object is immutable; a mutable root of trust provides an opportunity to spoof the actions of the DSC, thus threatening the foundation of trust that the DSC is intended to provide. For example, the data comprising the root of trust identity may be fused into the hardware of the DSC during manufacture. Therefore, it is imperative that the DSC is secure by design and that the foundation in which that security is established cannot be undermined.

### DSC Characteristics

The security functional requirements rely on the following characteristics of the DSC in terms of user, subjects, objects, security attributes, and operations.

Users: The entities using the DSC will be client applications on the platform. They may be acting as proxies for users, or they may have identities of their own. The DSC will not be able to distinguish the difference; therefore, the cPP will recognize an entity known as the CA, which stands for Client Application, as the user presenting authentication and authorization data to the DSC for the purposes of identity verification and authorization to perform operations. Section 1.4.1 discusses further the concept of users. This cPP also recognizes a special user called the administrator, which typically has access to DSC objects normally denied to CAs (see definition of objects below).

Subjects: The following list contains the fundamental actors in the expected operational use cases of the DSC. The first three are active actors, while the fourth may usually be passive but could be active.

* S.DSC – DSC with security attribute DSC.ID, which is the identity of the DSC
* S.Admin – Admin (an authorized administrator with special privileges) security attribute Admin.ID – See section 2.4.1 for more discussion on Admin.
* S.CA – CA or Client Application (i.e. an authorized user or an application with a verifiable identity) with security attribute CA.ID – See section 2.4.1 for more discussion on users.
* S.EPS – External Platform Storage or EPS (e.g. transient SDE/SDO source and destination, in the case of data imported and exported for the sole use inside the DSC). In the case of a passive EPS, the DSC will properly protect the integrity and confidentiality of the objects is stores there as well as retrieve from there. In the case of an active EPS with security attribute EPS.ID, the DSC and EPS may choose to create a secure channel through which they will pass objects back and forth.

Objects: The following list contains objects the DSC expects to use during the expected operational use cases.

* OB.P\_SDO – Pre-provisioned SDOs (e.g. DSC.ID) with security attributes listed in the next session.
* OB.T\_SDO – Transient SDOs or just SDOs (i.e. SDOs in the DSC currently, but are either ephemeral or are normally stored external to DSC when not in use) with security attributes listed in the next sessions.– See sections 2.4.2, 2.4.4, and 2.4.6 for more discussion on keys, which are the primary use cases for SDOs.
* OB.AuthData – Authorization Data (including authentication data, e.g. PINs, passwords)
* OB.PState – Platform State (e.g. measurements and assertions)
* OB.FAACntr – Failed Authorization Attempt Counters
* OB.AntiReplay – Anti-replay tokens (e.g. counters, nonces, etc.)
* OB.Context – Session Context (The DSC may maintain one or more sessions with a CA involving one or more of SDOs, Authorization Data, Platform State, Failed Authorization Counters, and Anti-Replay Tokens. The DSC may represent internally the state of these objects at any given time in a Session Context) – See Section 2.4.3 for more discussion on sessions.

Security attributes: The following list contains the minimum security attributes for a DSC. Individual DSCs may implement additional security attributes beyond this (whether they are additional standalone attributes or additional attributes that are associated with SDOs); the ST author is expected to identify these.

* DSC.ID – the DSC identifier. It may also serve as the identifier for the DSC Root of Trust
* CA.ID – the Client Application Identifier.
* EPS.ID – the External Platform Storage (EPS) identifier. This attribute is optional for a passive EPS (i.e. plain memory that only stores information). If the DSC uses a Client Application to manage storage, then support for this attribute is required.
* SDO.\* - the SDO security attributes:
  + SDO.ID – SDO Identifier
  + SDO.Type – SDO Type
  + SDO.AuthData – SDO Reference authorization data
  + SDO.Reauth – SDO re-authorization conditions
  + SDO.Conf – SDO Confidential SDE list
  + SDO.Export – SDO export flag
  + SDO.Integrity – SDO integrity protection data
  + SDO.Bind – SDO binding data

Operations: The following list contains the expected operations of a DSC.

* OP.Import (See Parse and Capture) – The DSC may receive SDOs, SDEs, Authorization Data, Platform State, Anti-Replay Tokens or Session Contexts from the CA or the EPS. The Admin may also give the DSC Authorization Data.
* OP.Create (See Provision and Compose) – The DSC may create SDOs, SDEs, Authorization Data, Platform State, Anti-Replay Tokens, or Session Contexts with authorization from a CA or Admin.
* OP.Use (See Process and Control) – The DSC may use or perform a cryptographic operation on Pre-provisioned SDOs, Transient SDOs, SDEs, Authorization Data, Platform State, Anti-Replay Tokens, or Session Contexts with Create authorization from a CA or Admin. Cryptographic operations may include encryption, decryption, hashing, signature generation, and signature verification.
* OP.Modify (See Process and Control) – The DSC may modify SDOs, SDEs, Authorization Data, Platform State, Anti-Replay Tokens, or Session Contexts with authorization from a CA or Admin
* OP.Attest (See Prove and Confirm) – The DSC may create an attestation of Platform State using an SDO or Pre-Provisioned SDO and Anti-Replay Tokens as authorized by a CA or Admin respectively.
* OP.Store (See Protect and Cushion) – The DSC may store SDOs, SDEs, Authorization Data, Platform State, Anti-Replay Tokens, or Session Contexts in protected storage of the DSC. See Section 2.4.5 for more discussion on protected storage.
* OP.Export (See Propagate) – The DSC may export SDOs, SDEs, Authorization Data, Platform State, or Anti-Replay Tokens to a CAs or EPS with the proper authorization from the owner of each object. In the case of EPS, the DSC will bind the objects to the DSC in such a way as to deny other DSCs or entities the ability to import, use, modify, attest, store, export, or destroy them. The DSC may export Session Contexts only to an EPS binding it in the same way as above.
* OP.Destroy (See Purge and Cleanse) – The DSC may purge SDOs, SDEs, Authorization Data, Platform State, Anti-Replay Tokens, or Session Contexts in protected storage with proper authorization from the owner of each object.

## TOE Use Cases

DSCs are used in platforms to support mobile commerce, to manage platform credentials, manage user access to sensitive resources such as enterprise data centers or entertainment content servers, to manage and protect data-in-transit such as through secure channels or VPN tunnels, and to manage and protect keying, authentication, and authorization material for data at rest solutions such as self-encrypting drives.

For the mobile commerce use case, users, merchants, and financial institutions expect and require that financial transactions between them and their platforms be trusted and secure. For example,

* All peer to a transaction must be able to authenticate each other.
* The integrity of the transaction must be ensured.

To support such transactions, a DSC ingests data elements and attributes and exports the data objects associated with these transactions and the identities of the parties. It generates data objects to use for these transactions. It securely stores data elements bound with their attributes within a protected hardware boundary. It authenticates and processes these data elements within a protected execution environment to ensure the authenticity of the parties and the transactions. It establishes secure communications channels between the parties to ensure the integrity and confidentiality of the transactions. It securely erases data objects when no longer needed. Lastly, a DSC ensures its own integrity and authenticity prior to execution. DSCs are implemented to satisfy the following use cases:

**[USE CASE 1] Protected Key Store**

A platform leveraging DSCs as a hardware-secured Private Key Store facilitates the use of secure and protected storage of asymmetric/symmetric private/secret keys for access to data and services. These DSCs would provide safe use of the private/secret keys inside the protected hardware boundary.

**[USE CASE 2] User / Platform Authentication to Enterprise Managed Resources**

A platform leveraging DSCs for a hardware-secured ID facilitates the use of the platform as a secure and reliable form of authentication for authorized access to highly sensitive local and/or remote data and services.

**[USE CASE 3] Mobile Commerce**

A platform leveraging DSCs facilitates secure storage and protected use of credentials for financial transactions between trusted and authorized users, platforms, merchants and financial institutions. These DSCs would provide safe use of the credentials inside the protected hardware boundary. The use of certified hardware-isolated credential stores on smart platforms and only unlocking their use with authenticated authorization provides confidence that the transaction was indeed authorized by the approved ‘platform holder’.

## Key Reference Model

The Key Reference Model abstraction draws inspiration from several different DSC products. The products distinguish themselves from one another in the types of keys supported, how they are protected, the types of applications supported, the number of layers of key, and the number of keys at each layer.

The following paragraphs describe the relationships between elements of the DSC.

### Roles

There are two main roles that come into play with any platform, including the DSC. These are administrative and user. The DSC is often a component within a larger system or platform that is referred to as a platform from this point forward. Often the platform supports different roles as well. At times, these roles may coincide with the roles supported in the DSC, even on purpose.

The administrative role may be further divided into manufacturer administrator and owner administrator. The manufacturer administrator may, among other things, accept responsibility for providing timely updates to the DSC, both feature updates and security updates. It may also be responsible for managing the pre-installed SDOs and the initial configuration of the DSC. However, the manufacturer administrator is not expected to take an active part in managing the contents of the DSC that the owner administrator or some user may have installed post-manufacture.

The owner administrator, on the other hand, may manage the contents of the DSC, including managing user content. The role of owner administrator of the DSC may coincide with the role of the owner of the platform that contains the DSC. Often the platform owner is more commonly referred to as the system administrator, or just the administrator. The concept is more or less identical. However, we should try not to confuse a DSC owner administrator or platform owner administrator with the owner of an SDO. Although the DSC or platform owner administrator may possibly own one or more SDOs, not all SDOs allow the DSC owner administrator direct control of it. In some cases, the DSC owner administrator may also be in a position to grant or deny the manufacturer administrator access to what it perceives as their content, namely its firmware and possibly some keying material belonging to the manufacturer. The manufacturer’s choice of allowing the owner administrator of the DSC this kind of latitude is a feature of its product.

The other role, that of the user, may also be further divided into multiple users. One user could be an application vendor acting on its own behalf to update software on the platform. Another user could be a content provider gating access to its content through an application. Another user could be a user of the platform with an identity it uses to authenticate themselves to the content provider through an application. Another user could be an original equipment manufacturer (OEM) which designed and manufactured a more complex system with the DSC as a component (assuming that the DSC manufacturer and the manufacturer of the more complex system using the DSC as a component are different entities). In some cases, the DSC may allow the OEM to provision and manage its own content in the DSC for its own purpose, such as managing their firmware or software installed on the platform. In this case, we consider the OEM as another user under the control of the owner administrator. We do not ascribe the role of administrator to the OEM since it likely does not control the manufacturer’s firmware or key material, and thus does not control the behavior of the DSC. Nor would the other users of the platform tolerate OEM control of their content stored in the DSC. Even so, there should be some separation between the administrator-owner and the other roles of the platform in terms of authorizing use of the contents assigned to each of the roles. I.e., administrator-owners may deny access to contents, either temporarily or permanently (e.g. through cryptographic erase). However, they cannot themselves access their contents for their own use or to gain access to things they are not otherwise authorized to access.

### Key Usage

One way to categorize keys is by the cryptographic functions they are allowed to participate in. When one creates a key, one often restricts its use to encryption and decryption, or to signature generation and verification. There are exceptions to this rule, especially in proof of possession protocols. However, certification regimes often require strict separation of usage in regards to encryption/decryption and signature generation/verification: one may use a key for one or the other, but never both. As such, a DSC may have to enforce this separation of usage for keys; this may mean that an attribute must accompany a key to help the DSC in its enforcement.

### Sessions

Users may use their keys multiple times while in the DSC. The DSC may enforce authorization each time a user uses their keys within it. Authorization using public key methods tend to be resource intensive (i.e. uses a fair amount of internal memory and takes a long time). As an alternative the DSC could allow the user/owner of the key to open a session and provide the authorization once the first time it uses it, then maintain the session and authorization using a series of less resource intensive challenges and responses. Such a protocol of challenges and responses may generate and use ephemeral authorization tokens, which would be one form of critical security parameters (CSPs). The DSC may have to switch session contexts in and out of the DSC to external temporary storage, which necessitates the protection of these CSPs. (Such session contexts is one type of SDO, to be discussed later.)

### Key Hierarchies

Another way to categorize keys is the relationship they have with each other. This model calls out three categories of keys generally found on typical DSCs. DSCs may contain Root Keys, Intermediate (or Branch) Keys, and Leaf Keys.

Most DSCs have a concept of Root Keys. These keys are typically provisioned by the DSC manufacturer and have some permanence in the DSC. Root Keys may be derived from seeds (which we discuss later), injected at manufacturing time, or provisioned by a user. Root keys installed by the manufacturers are considered manufacturer administrator key material. I.e., typically normal users, including OEMs, should not alter or erase this material, unless specifically authorized to do so. Root keys installed by the owner administrator should be similarly restricted. User-installed root keys, on the other hand, are not considered as permanent since the user or the owner administrator can remove them at any time without user authorization.

Root Keys may either be Encryption/Decryption Keys, Signature Verification Keys or Signature Generation Keys. Encryption/decryption keys, or simply Root Encryption Key (REK), usually anchor a hierarchy of keys stored external to the DSC necessitating both the encrypt key to protect the key outside the DSC, and the decrypt key to expose its operations within the protected and secure confines of the DSC. The Signature Verification Keys from public key schemes should always contain the public portion and never the private portion. Use of Signature Generation keys as Root Keys is rare.

Most DSCs have a concept of Intermediate Keys. These are sometimes known as Branch Keys, Key Encryption Keys, and Key Wrapping Keys. In the SFRs of this cPP, we will call them Key Encryption Keys (KEKs), even if the target of encryption is not a key. Intermediate Keys must always be encryption keys. Intermediate keys cannot be signing keys.

Note that although chained certificates (see certificates below) are one form of a sequence of keys, each of which signs another key, the creation and verification of such a chain of certificates is out of scope for the core requirements of the cPP and may be added as a package if one or both of these features (creating the chain and verifying the chain) is indeed present in the DSC. Nonetheless, the primitives of signing and verification are present due to other cryptographic operations in scope for this cPP.

Intermediate Keys should always be protected (i.e. wrapped) by either a Root Key or another Intermediate Key.

Leaf Keys can be either signing or encryption keys. In fact, we are going to broaden the scope of Leaves to include Authorization Tokens. All Leaf Objects (including Keys and Authorization Tokens) should be wrapped by either a Root Key or a KEK. Encryption Leaf Keys do not wrap other keys (at least in the context of the DSC; what happens outside the DSC with Leaf Keys is out of its control). In many contexts, an Encryption Leaf Key is known as a Data Encryption Key (DEK). In the context of the DSC, this cPP will not assume how the user of the DSC will utilize the Leaf Keys it creates, and will refrain from using the term DEK.

Certificates contain either signed public keys, signed encryption/decryption key, or some sort of authorization token or authentication token. Signature keys come in several varieties: asymmetric signing keys, which contains a private key for signing (and maybe also the public key for verification) and verification keys, which contains only the public verification key and does not contain the private key (and thus cannot perform a signing function). There are also symmetric signature keys. In this case these consist of only a single key for both signing and verifying.

An Authorization Token is an arbitrary length of bits. These bits can either be random or non-random. The contents of an Authorization Token only make sense to the entity consuming it.

Seeds have a special place in this Key Reference Model. There are permanent seeds manufacturers, owners, and users of the DSC can use to create root keys. Manufacturers have good reasons to use seeds to derive Root Keys and other items in the Key Reference Model. These include:

* Seeds take less space to store than certain asymmetric keys for given desired cryptographic strengths.
* Seeds are not keys, thus may not subject to key zeroization requirements in FIPS 140-2.
* Seeds that are unique per DSC enhances the chance that the same key derivation function on different DSCs will yield unique keys.

Figure 5 contains an example of a hierarchy of keys where each lower level key is wrapped by a higher-level key that is connected to it. The Firmware Signature Key and the Root Encryption Key are examples of Root Keys. The Intermediate Wrapping Key is an example of an Intermediate Key. The Software Signature Key, the File Encryption Key, and the Streaming Movie Authorization Token are examples of Leaf Objects. Figure 5 serves as an illustration of key hierarchies; other configurations are possible.

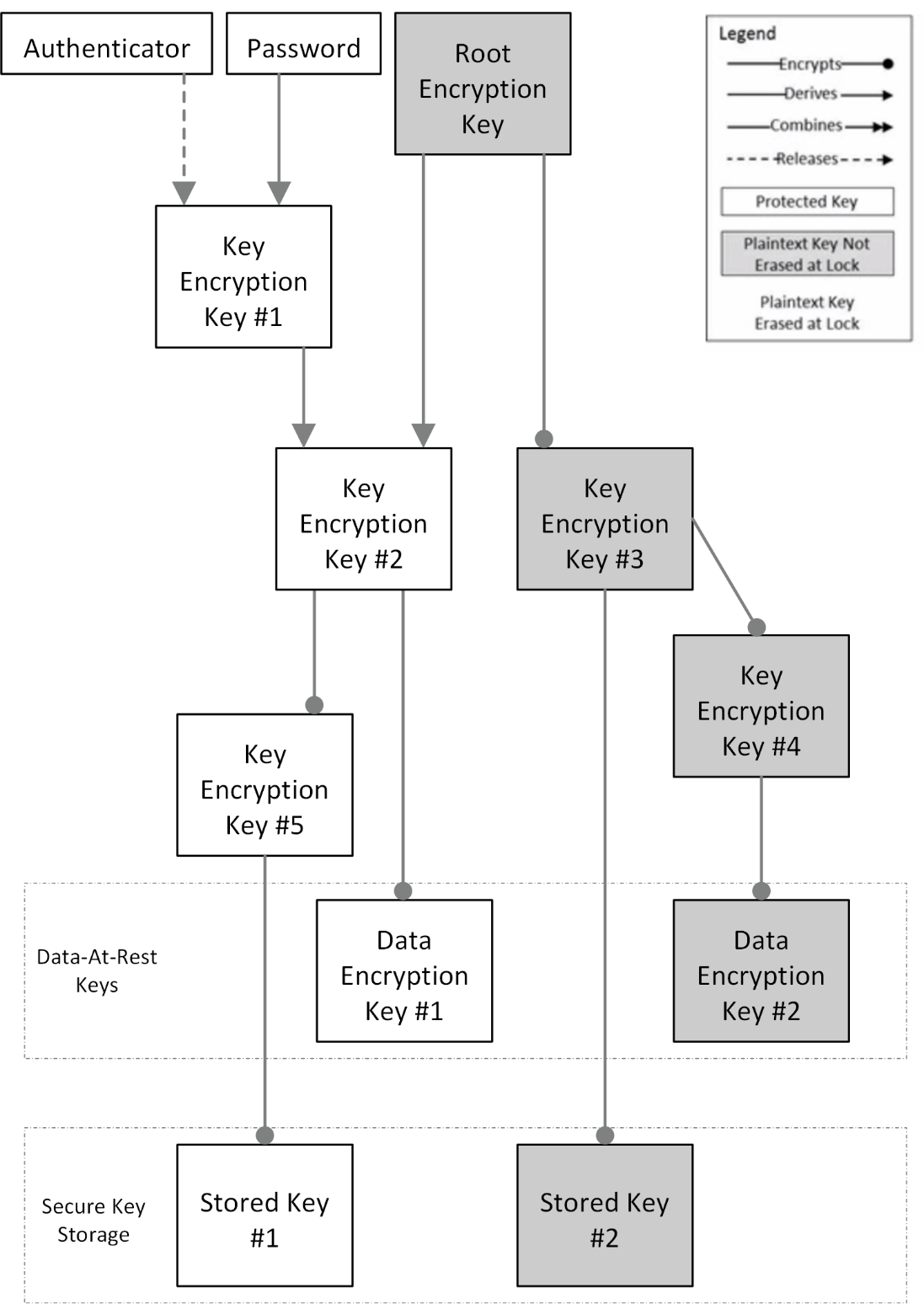


Figure 5: Example Key Hierarchy

Roles may play an important part in key hierarchies. One of the simplest model enforces a different hierarchy for each role at the root key level. Another way to put this is each hierarchy at the root key level supports a different role. However, for more complexity, once we allow intermediate keys, then each intermediate key could serve as the root of a hierarchy of keys for a different role. Here is where the key functions and the roles come together. User roles may further divide into which role has the right to use a key, which role has the right to move the key from one parent to another, which role has the right to destroy a key, etc.

### Protected Storage Locations

This cPP covers several different types of storage locations for keys and critical security parameters (CSPs) such as authorization tokens. Some DSCs may have a generous amount of protected storage internal to themselves, which allows it to accommodate all keys and CSPs in operational use, whether the DSC is performing operations to administer itself or operations on behalf of users. Other DSCs may have a minimal amount of protected storage locations with just enough to accommodate root keys along with a limited number of operational keys and CSPs for user authorized sessions.

For those cases in which the DSC relies on storage external to itself to accommodate all the keys and CSPs on which applications expect it to operate, it will either have to support secure channels to another DSC with a more generous allocation of protected storage locations, or use a series of wrapping keys to protect private keys and CSPs while outside of the DSC. Whether the DSC is powered on or powered off, the DSC is expected to provide support for protected storage locations for its root keys. If the DSC utilizes external storage without secure channels, then it should be ready to wrap both intermediate wrapping keys as well as the leaf objects. This implies that there will be some sort of structure on each of these items stored external to the DSC. The next section discusses that structure.

### SDEs and SDOs

Although there is another section written about Security Data Elements (SDEs) and Security Data Objects (SDOs), here we want to map keys and authorization tokens to SDEs and SDOs. This cPP does not impose a strict structure on the items in the key hierarchy. An X.509 certificate is one example of a strict structure of a key with attributes. Collecting attributes of an SDE and composing an SDO structure with an SDE and attribute fields imposes temporal and storage penalties in all cases. In certain resource-constrained cases the attributes could be implicit. For example, the root keys are administrative keys, which requires administrator authorization for use while all other objects are user objects, which require user authorization. The raw unadorned key or object is the SDE and the SDO may be implied by virtue of its location within the hierarchy, i.e. it is understood that keys in the root position require administrator authorization while all other objects, which may or may not be keys, require user or administrator authorization.

In the previous section on protected storage locations, a DSC may have to utilize storage external to itself. In these cases, an SDO of a wrapped key may contain a number of important attributes, such as a pointer to its parent, authorization values, and other indications of the functions allowed (encrypt vs. sign). Alternatively, some or all attributes may be implied, which means that only the keys or CSPs themselves exist outside the DSC. In either case, the sensitive values, such as private keys, secret keys, and CSPs, should be encrypted when outside the DSC. The parent of these objects are either intermediate wrapping keys, or encrypting root keys.

Some DSCs may want to distinguish between SDEs created within itself from SDEs ingested from an external source. Additionally, some DSCs may output SDEs without additional context or attributes from the DSC. A DSC, in some contexts, will not distinguish an ingested SDO from raw keys.

# CC Conformance Claims

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

* conforms to the requirements of Common Criteria v3.1, Release 5
* is Part 2 extended, Part 3 conformant
* does not claim conformance to any other PP.

The methodology applied for the cPP evaluation is defined in [CEM] and refined by the Evaluation Activities in [SD]. This cPP satisfies the following Assurance Families: APE\_CCL.1, APE\_ECD.1, APE\_INT.1, APE\_OBJ.2, APE\_REQ.2 and APE\_SPD.1.

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

# Security Problem Definition

## Assets

(R.SDE) A security data element is an item of user data that is held in (and may be stored on) the DSC and that may be used only by an authorized subject (i.e. a user or process acting on behalf of that user). Typically the DSC will not know what a security data element represents in terms of the application or service that it is used for: it will characterize a security data element only in terms of the authorization requirements that are necessary to access it (i.e. the presentation and possibly processing of an authorization factor presented to the TOE), and the operations that can be performed on it after authorization has been achieved. A Security Data Element may require protection of its confidentiality, its integrity, or both.

(R.SDO) A security data object comprises one or more security data elements that are collectively bound to one or more attributes (e.g. an identifier for the identity that a key or authorization factor is associated with). These attributes may necessarily be used by the DSC to enforce authorization policies concerning the allowed use and disposition of the subject SDE(s). The bindings can either be explicit (e.g. in a well-formatted standards-based data structure) or implicit (e.g. by virtue of their location within the DSC which implies privileges of use and disposition by certain users), or some combination of the two.

(R.CONFKEY) Confidential (or secret) keys used in symmetric cryptographic functions and private keys used in asymmetric cryptographic functions are managed and used by the TOE in support of the cryptographic services that it offers. This includes user keys that are owned and used by a specific user (which are a special case of a Security Data Element), and support keys used in the implementation and operation of the TOE. The confidentiality and integrity of these keys must be protected**.**

(R.PUBKEY) Public keys are managed and used by the TOE in support of the cryptographic services that it offers (including user keys and support keys). This includes user keys that are owned and used by a specific user (which are a special case of a Security Data Element), and support keys used in the implementation and operation of the TOE. The integrity of these keys must be protected.

(R.AUTHTOKEN) Authorization Tokens are managed and used by the TOE in support of the authorization services that it offers, including both user provided authorization tokens and those created by the DSC. Authorization Tokens may be special cases of SDEs, and/or they may be attributes in an SDO. The DSC may use Authorization Tokens to manage the use and disposition of a single SDE, or a broad class of SDEs. The DSC protects the integrity of Authorization Tokens, and in some cases, may protect their confidentiality.

## Threats

(T.UNAUTHORIZED\_ACCESS) An adversary may gain unauthorized access to one or more security data elements within the DSC. If an adversary gains access to SDEs/SDOs stored in the TSF, they may attempt to view, use, or destroy this data as well as impersonate a user or that user’s platform.

The consequences of unauthorized access to SDEs/SDOs include the loss of confidentiality of their content, unauthorized use of that content, unauthorized modification or destruction of that content, and the ability of the adversary to impersonate a user or that user’s platform.

Rationale:

* O.AUTHORIZATION defines and enforces policies that govern access to SDOs.
* O.DATA\_PROTECTION ensures that SDOs have adequate protections.

(T.SDE\_TRANSIT\_COMPROMISE) An adversary may access or determine plaintext values of keys, authorization data, and other security data elements as the DSC transmits them into or out of the DSC. This puts the SDE/SDO data, user identity, and the TOE’s underlying platform at risk.

The consequences of access to plaintext security data elements in this way include the loss of confidentiality of SDE/SDO data, unauthorized use of this data, unauthorized modification this data, and the ability of the adversary to impersonate a user or their platform.

Rationale:

* O.PARSE\_PROTECTION ensures that SDEs are not transmitted into the DSC over an insecure channel.
* O.PURGE\_PROTECTION ensures that residual data associated with SDEs do not remain when the SDEs themselves are deleted.
* O.DATA\_PROTECTION ensures that the confidentiality of SDEs is enforced.

(T.WEAK\_OWNERSHIP\_BINDING) A user may successfully access or manipulate security data elements that they do not own.

The consequences of manipulation of security data elements in this way include the loss of confidentiality of SDE/SDO data, unauthorized use of that data, unauthorized modification of that data, and the ability of the adversary to impersonate a user or that user’s platform.

Rationale:

* O.DATA\_PROTECTION protects SDEs from unauthorized access.
* O.STRONG\_BINDING establishes ownership of SDEs to determine the users that may interact with them.
* O.FW\_INTEGRITY ensures that the DSC’s firmware cannot be corrupted in a way that causes ownership bindings not to be enforced.

(T.WEAK\_ELEMENT\_BINDING) An adversary may successfully break the association between security data elements, for example to replace one element with another element.

The consequences of manipulation of security data elements in this way include the loss of confidentiality of the data, unauthorized use of the data, destruction of the data, unauthorized modification of credentials, and the ability of the adversary to impersonate a user or that user’s platform.

Rationale:

* O.DATA\_PROTECTION assures the authenticity and integrity of SDEs.
* O.FW\_INTEGRITY ensures that the DSC’s firmware cannot be corrupted in a way that allows the unauthorized substitution of SDEs.

(T.BRUTE\_FORCE\_AUTH) Threat agents may hammer the DSC repeatedly with authorization factors, such as passwords, biometrics, etc. that protect the security data elements. Successful hammering puts the SDE/SDO data, user identity, and the TOE’s underlying platform at risk.

The consequences of risks to security data elements include the loss of confidentiality of the SDE/SDO data, unauthorized access to and use of this data, destruction of this data, and the ability of the adversary to impersonate a user or that user’s platform.

Rationale:

* O.AUTH\_FAILURES ensures that the DSC has a method to thwart brute-force authentication attempts.

(T.WEAK\_CRYPTO) Threat agents may cryptographically exploit poorly chosen cryptographic algorithms, random bit generators, ciphers or key sizes. Weak cryptography chosen by users or by TSF protection mechanisms puts the user’s data, identity, and platform at risk of exploitation by adversaries.

The consequences of risks to security data elements include the loss of confidentiality of the SDE/SDO data, unauthorized access to and use of this data, destruction of this data, and the ability of the adversary to impersonate a user or that user’s platform.

Rationale:

* O.STRONG\_CRYPTO ensures that the DSC implements cryptographic algorithms that are not subject to compromise.

(T.UNAUTH\_UPDATE) Threat agents may force the platform to update the DSC with firmware that compromises its security features. Poorly chosen update protocols, cryptographic algorithms, and keys sizes may allow adversaries to install software and/or firmware that bypasses security features or rolls back to firmware versions with compromised security features and provides them with unauthorized access to security data elements.

The consequences of risks to firmware include the loss of confidentiality of the SDE/SDO data, unauthorized access to and use of this data, destruction of this data, and the ability of the adversary to impersonate a user or that user’s platform.

Rationale:

* O.SECURE\_UPDATE prevents the application of untrusted firmware updates to the DSC.

(T.HW\_ATTACK) An adversary may apply hardware attacks such as probing, physical manipulation, fault-injection, side-channel analysis, environmental stress, or reactivating blocked test-features or other pre-delivery services to manipulate the behavior of the TOE to disclose SDOs.

Rationale:

* O.PURGE\_PROTECTION ensures that a hardware attack does not expose SDE remnants that could compromise the DSC or any of its stored data.
* O.AUTHORIZATION ensures that the access control policy is not thwarted by physical attacks on the DSC.

## Assumptions

This section describes the assumptions made in identification of the threats and security requirements for dedicated security components. The dedicated security component is not expected to provide assurance in any of these areas, and as a result, requirements are not included to mitigate the threats associated.

(A.CREDENTIAL\_REVOCATION) If a platform is lost, stolen, or compromised then there is a method of revocation of any credentials held (or equivalent method of mitigating the impact of potential access to the credentials). Credential revocation ensures that the loss of physical custody does not have significant negative impact on the security of the platform. This implies that an attacker has only limited access to the device to apply attacks. It further implies that the device owner is not seen as an attacker.

Rationale:

* OE.PHYSICAL ensures that an adversary will not have sufficient access to the TOE to exploit the login mechanism if the assumption holds that credential revocation is enforced upon a lost or stolen TOE.

(A.AUTH\_USERS) Authorized users follow all provided guidance regarding the safeguarding of security data element(s) held outside the TOE.

Rationale:

* OE.AUTH\_USERS holds that sufficiently trained and trusted users will follow instructions as assumed.

(A.TRUSTED\_PEER) The remote peer communicating over a secure channel is trustworthy, and will not abuse the secure channel in order to introduce malware or fraudulent security data elements into the DSC.

Rationale:

* OE.TRUSTED\_PEER holds that if the TOE’s Operational Environment is configured such that the TSF can only communicate with trusted peer, then this assumption will be satisfied.

(A.ROT\_INTEGRITY) The vendor provides a Root of Trust (RoT), comprising of the DSC firmware, hardware, and pre-installed Security Data Objects (SDOs), free of intentionally malicious capabilities. The platform trusts the RoT since it cannot verify the integrity and authenticity of the RoT.

If the RoT is immutable, then the platform can have confidence that once delivered, malicious actors cannot modify the RoT to add malicious capabilities. If the RoT is mutable (e.g. the firmware and pre-installed SDOs), then it will verify the authenticity and integrity of the updates before applying them.

Rationale:

* OE.TRUSTED\_PEER holds that the vendor’s Root of Trust can be relied upon if the only entities that the TSF communicates with are trusted.

## Organizational Security Policies

There are no organizational security policies defined in this cPP.

# Security Objectives

## Security Objectives for the TOE

(O.AUTHORIZATION)The TOE ensures that only authorized subjects can access Security Data Objects stored by the users of the DSC, pre-installed SDOs stored in the RoT by the manufacturer of the DSC, and management functions that are used to manipulate the DSC and its stored data.

Rationale:

* FCS\_STG\_EXT.1 ensures that key data is placed into secure key storage and cannot be modified by untrusted subjects.
* FDP\_ACC.1 defines an access control policy that governs the authorization required to interact with SDOs.
* FDP\_ACF.1 defines the rules enforced by the access control policy defined in FDP\_ACC.1 to control access to SDOs.
* FDP\_ETC\_EXT.1 ensures that protected data propagated outside the TOE is not disclosed to any unauthorized subjects.
* FDP\_ITC.1 ensures that imported data that lacks security attributes is handled by the TSF in such a way that appropriate access can be controlled to the data once stored inside the TOE boundary.
* FDP\_ITC.2 ensures that imported data that contains security attributes have those attributes maintained once stored inside the TOE boundary, which can be used to enforce access control rules against the data.
* FIA\_UIA\_EXT.1 defines the methods by which users authenticate to the TOE to prove their identity prior to interacting with any protected data.
* FIA\_UAU.6/SDO defines when authorization checks are performed for user requests to access SDOs.
* FMT\_MOF\_EXT.1 enforces access control on the management functions provided by the TOE.
* FMT\_MSA.1/SDO enforces restrictions on the subjects that can interact with SDOs and their attributes.
* FMT\_MSA.3/SDO defines the default access restrictions that are enforced on SDO attributes if not overridden by specific access control policy rules.
* FMT\_SMF.1 defines the management functions that are provided by the TOE to authorized subjects.
* FMT\_SMR.2 defines the roles used by the TSF for enforcement of access control to protected functions and data.
* FPT\_FLS.1/FI ensures that fault injections cannot be used to circumvent access control policy restrictions preventing a user from accessing protected functions or data.
* FPT\_MOD\_EXT.1 ensures that there are no accessible debug modes that could be used to circumvent access control policy restrictions preventing a user from accessing protected functions or data.
* FPT\_PHP.3 ensures that some mechanism is in place to thwart unauthorized attempts to access protected functions or data through physical tampering of the TOE.
* FPT\_PRO\_EXT.1 defines the Root of Trust for the TOE, which is used to derive all access control functionality.
* FPT\_ROT\_EXT.2 enforces the Root of Trust for Storage to enforce access control against SDOs.
* FRU\_FLT.1 ensures that fault injection attempts do not interfere with the enforcement of access control against protected data.
* (selection-based) FIA\_AFL\_EXT.2 defines the access control that is enforced on an SDO if excessive authentication failures block access to it.
* (selection-based) FIA\_UAU.5/Prove provides the TSF with the ability to specify the use of multiple authentication mechanisms as a prerequisite to granting access to protected functions or data.

(O.PARSE\_PROTECTION)All security data elements are received by the TOE over a secure channel for parsing, protecting confidentiality and integrity of the security data elements while in transit. The TOE authenticates the source of all security data elements received, and authenticates itself to the remote peer.

Rationale:

* FDP\_APW\_EXT.1 ensures that the confidentiality of authentication-related data is protected prior to storage.
* FDP\_IFC.1 defines an information flow control policy that governs the secure creation, parsing, and propagation of SDOs.
* FDP\_IFF.1 defines the rules enforced by the information flow control policy defined in FDP\_IFC.1 to ensure that parsed data is protected.
* FDP\_ITC\_EXT.1 ensures that all SDEs parsed by the TOE are transmitted over a secure channel.
* FDP\_ITC\_EXT.2 ensures that all SDOs parsed by the TOE are transmitted over a secure channel.
* (selection-based) FTP\_ITE\_EXT.1 defines the cryptographic method used to transfer data between the TOE and external entities.
* (selection-based) FTP\_ITP\_EXT.1 defines a physically protected channel that the TSF can use to securely parse data being imported into it.
* (selection-based) FTP\_ITC\_EXT.1 defines a cryptographically protected channel that the TSF can use to securely parse data being imported into it.

(O.PURGE\_PROTECTION)The TOE provides secure destruction of security data elements when they are deleted, so that the previous value of the security data element can no longer be accessed (and cannot be restored).

Rationale:

* FCS\_CKM.4 ensures that key data is destroyed in a manner that prevents its future recovery.
* FCS\_CKM\_EXT.4 ensures that key data is not retained for any longer than necessary for its intended usage.
* FDP\_FRS\_EXT.1 defines the condition in which a factory reset will be initiated, which triggers a purge of stored SDEs.
* FDP\_RIP.1 ensures that any purged SDEs/SDOs are erased in residual memory so that their future recovery is prevented.
* (selection-based) FDP\_FRS\_EXT.2 ensures that all user-specific SDOs are purged upon factory reset and may indicate any factory default SDOs that are reset to their initial values.

(O.DATA\_PROTECTION) The TOE provides authenticity, confidentiality, and integrity services for security data objects.

Rationale:

* FCS\_COP.1/Hash provides a cryptographic operation for asserting the integrity of SDOs.
* FCS\_COP.1/HMAC provides a cryptographic operation for asserting the authenticity of SDOs.
* FCS\_COP.1/SigGen provides a cryptographic operation for preserving the authenticity of SDOs.
* FCS\_COP.1/SigVer provides a cryptographic operation for asserting the authenticity of SDOs.
* FCS\_COP.1/SKC provides a cryptographic operation for maintaining the confidentiality of SDOs.
* FCS\_STG\_EXT.1 ensures that key data is placed into secure key storage and cannot be modified by untrusted subjects.
* FCS\_STG\_EXT.2 ensures that confidentiality of key storage is maintained using strong cryptography.
* FCS\_STG\_EXT.3 ensures that integrity of key storage is maintained using strong cryptography.
* FDP\_APW\_EXT.1 ensures that the confidentiality of authentication-related data is protected prior to storage.
* FDP\_ETC\_EXT.1 ensures that the confidentiality of protected data propagated outside the TOE is maintained.
* FDP\_IFC.1 defines an information flow control policy that governs the secure creation, parsing, and propagation of SDOs.
* FDP\_IFF.1 defines the rules enforced by the information flow control policy defined in FDP\_IFC.1 to ensure that propagated data is protected.
* FDP\_ITC\_EXT.1 ensures that all SDEs parsed by the TOE have verifiable integrity.
* FDP\_ITC\_EXT.2 ensures that all SDOs parsed by the TOE have verifiable integrity.
* FDP\_SDC\_EXT.1 ensures that SDEs/SDOs are stored with confidentiality.
* FDP\_SDI.2 ensures that SDEs/SDOs are monitored for integrity violations.
* FPT\_APW\_EXT.1 protects authentication data from unauthorized disclosure.
* FPT\_ROT\_EXT.1 defines the Root of Trust services that are available for the protection of data.
* FPT\_RPL\_EXT.1 ensures that access control restrictions cannot be bypassed through replay of operations.
* (optional) FPT\_PRO\_EXT.2 ensures that the TSF can produce attestation of the integrity of its stored data.
* (optional) FPT\_ROT\_EXT.3 allows the TSF to provide a Root of Trust for Reporting that can provide assured information about the stored SDEs.
* (selection-based) FDP\_DAU.1/Prove defines the Prove service that can be used to invoke the Roots of Trust for Measurement and Reporting and provide affirmation of the validity of stored data.
* (selection-based) FPT\_ITT.1 ensures that confidentiality and integrity is maintained in cases where data is transmitted between physically separate parts of a distributed DSC.

(O.STRONG\_BINDING)The TOE provides a mechanism for binding data to its attributes (including the identity of its owner) and prevents unauthorized changes to data attributes.

Rationale:

* FDP\_ITC\_EXT.1 ensures that all SDEs parsed by the TOE include appropriate binding metadata.
* FDP\_ITC.2 ensures that imported data that contains security attributes have those attributes maintained once stored inside the TOE boundary, which may include binding data.
* FIA\_USB\_EXT.1 defines the TSF behavior that is enforced against SDOs upon the binding of a user to access the SDOs.

The protections for pre-installed SDEs/SDOs come through the firmware protections. I.e. only authorized updates to the firmware contains the functionality that determine the attributes of the pre-installed SDOs. In the same vein, the authorized updates may also affect the SDEs as well, if the vendor so chooses. The authorized update binds the attributes present in the functionality of the firmware to the pre-installed SDEs.

(O.AUTH\_FAILURES)The TOE resists repeated attempts to guess authorization factors by responding to consecutive failed attempts in a way that prevents an attacker from exploring a significant amount of the space of possible authorization data values.

Rationale:

* FIA\_AFL\_EXT.1 enforces authentication failure handling capabilities to ensure that brute force attacks on the TSF are not possible.
* FIA\_SOS.2 protects against brute force authentication by generating secrets that are statistically impossible to guess.
* FPT\_STM.1 provides reliable system time services that may be used to determine when excessive authentication failure attempts have been made.
* (selection-based) FIA\_AFL\_EXT.2 defines how access to an SDO is restored if excessive authentication failures trigger a lock on it.

(O.STRONG\_CRYPTO)The TOE implements strong cryptographic mechanisms and algorithms according to recognized standards, including support for random bit generation based on recognized standards and a source of sufficient entropy. The TOE uses key sizes that are recognized as providing sufficient resistance to current attack capabilities.

Rationale:

* FCS\_CKM.1 specifies the supported methods of key generation.
* FCS\_CKM.1/AK ensures the generation of strong asymmetric keys.
* FCS\_CKM.1/KEK ensures the generation of strong key encryption keys.
* FCS\_CKM.2 ensures the use of strong key establishment mechanisms.
* FCS\_CKM\_EXT.5 ensures the use of strong mechanism to perform key derivation.
* FCS\_COP.1/Hash ensures the use of strong hash mechanisms.
* FCS\_COP.1/HMAC ensures the use of strong HMAC mechanisms.
* FCS\_COP.1/KAT ensures the use of strong methods to perform key agreement and key transport.
* FCS\_COP.1/KeyEnc ensures the use of strong methods to perform key encryption.
* FCS\_COP.1/PBKDF ensures the use of strong methods to derive keys from password data.
* FCS\_COP.1/SigGen ensures the use of strong digital signature services.
* FCS\_COP.1/SigVer ensures the use of strong digital signature services.
* FCS\_COP.1/SKC ensures the use of strong methods to encrypt sensitive data.
* FCS\_RBG\_EXT.1 ensures the use of strong random bit generation mechanisms.
* FCS\_SLT\_EXT.1 ensures that salts and nonces used by the TOE do not negatively impact key strength.
* FCS\_STG\_EXT.2 ensures that confidentiality of key storage is maintained using strong cryptography.
* FCS\_STG\_EXT.3 ensures that integrity of key storage is maintained using strong cryptography.
* FPT\_STM.1 provides reliable system time services that may be used as inputs to cryptographic functions.
* (optional) FCS\_ENT\_EXT.1 provides an interface to access entropy data so that the TSF can support the use of strong cryptography in its operational environment.
* (optional) FCS\_RBG\_EXT.2 provides an external interface to seed the random bit generator that enforces strong cryptography by requiring a minimum amount of input.
* (selection-based) FCS\_CKM.1/SK ensures the generation of strong symmetric keys.
* (selection-based) FTP\_CCMP\_EXT.1 defines the implementation of CCMP (IEEE 802.11i) using strong cryptography.
* (selection-based) FTP\_GCMP\_EXT.1 defines the implementation of GCMP (IEEE 802.11ad) using strong cryptography.
* (selection-based) FTP\_ITE\_EXT.1 defines the cryptographic method used to transfer data between the TOE and external entities.

(O.SECURE\_UPDATE)The TOE software/firmware either does not allow update, or else implements a mechanism that ensures only authorized updates are applied. If the TOE allows updating its firmware, it is required to implement a mechanism that ensures only authorized firmware can be loaded into the TOE. A secure update mechanism ensures the firmware is authorized through verification of its integrity, authenticity while also preventing rollback to a previous and potentially vulnerable firmware instance.

Rationale:

* FDP\_MFW\_EXT.1 specifies whether the TOE’s firmware is mutable or immutable.
* (selection-based) FPT\_FLS.1/FW requires the TSF to take action to preserve its secure operation if a rollback attempt or invalid firmware update is detected.
* (selection-based) FPT\_RPL.1/Rollback ensures that the TSF will not permit rollback attempts of its firmware.

(O.FW\_INTEGRITY) The TOE ensures its own integrity has remained intact and attests its integrity to outside parties on request.

Rationale:

* FDP\_MFW\_EXT.1 specifies whether the TOE’s firmware is mutable or immutable, to determine the extent to which this is objective must be satisfied by other SFRs.
* FPT\_ROT\_EXT.1 defines the Root of Trust services that are available in the TOE, which can include Roots of Trust for measurement and reporting.
* FPT\_TST.1 defines the mechanisms used to verify and attest to the integrity of the TSF.
* (selection-based) FDP\_DAU.1/Prove defines the Prove service that can be used to invoke the Roots of Trust for Measurement and Reporting and provide affirmation of the validity of the TSF.
* (selection-based) FDP\_MPW\_EXT.2 ensures that the TSF can generate evidence that its mutable firmware integrity remains intact.
* (selection-based) FPT\_FLS.1/FW requires the TSF to take action to preserve its secure operation if any violations to its firmware integrity are detected.

## 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.PHYSICAL) The platform holder will ensure that an attacker has no prolonged, unsupervised physical access to the platform. If a platform is lost or stolen then the platform holder will promptly initiate revocation of any credentials held (or equivalent method of mitigating the impact of potential access to the credentials).

This security objective for the operating environment expects an entity to wipe the contents of the DSC in the event that an attacker has prolonged unsupervised physical access to the platform containing the DSC. There exists a variety of methods to wipe the contents or render the contents useless to the attacker. The platform may institute its own signal to wipe the DSC upon reaching or exceeding a threshold of unsuccessful user authentication attempts by an attacker. A remote entity may signal to the platform that it should issue a signal to the DSC to wipe is contents. The platform user (who has lost physical access to the platform) may contact service providers and inform them of the loss of credentials in the DSC, who may in turn issue revocation of those credentials. This method is much like calling credit card companies to report lost or stolen credit cards.

(OE.AUTH\_USERS) Authorized users follow all provided guidance regarding the safeguarding of security data element(s), especially authorization tokens such as passwords, pass-phrases, and biometrics.

(OE.TRUSTED\_PEER) Connections using secure channels are made only to trusted peers, in whom confidence has been established that they will not abuse the secure channel in order to introduce malware or fraudulent security data elements into the DSC.

## Security Objectives Rationale

Table 2 shows the mapping of Threats and Assumptions to Security Objectives for the TOE and for its Operational Environment. Rationale for these mappings is provided in the sections above.

|  | O.AUTHORIZATION | O.PARSE\_PROTECTION | O.PURGE\_PROTECTION | O.DATA\_PROTECTION | O.STRONG\_BINDING | O.FW\_INTEGRITY | O.AUTH\_FAILURES | O.STRONG\_CRYPTO | O.SECURE\_UPDATE | OE.PHYSICAL | OE.AUTH\_USERS | OE.TRUSTED\_PEER |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| T.UNAUTHORIZED\_ACCESS | X |  |  | X |  |  |  |  |  |  |  |  |
| T.SDE\_TRANSIT\_COMPROMISE |  | X | X | X |  |  |  |  |  |  |  |  |
| T.WEAK\_OWNERSHIP\_BINDING |  |  |  | X | X | X |  |  |  |  |  |  |
| T.WEAK\_ELEMENT\_BINDING |  |  |  | X |  | X |  |  |  |  |  |  |
| T.BRUTE\_FORCE\_AUTH |  |  |  |  |  |  | X |  |  |  |  |  |
| T.WEAK\_CRYPTO |  |  |  |  |  |  |  | X |  |  |  |  |
| T.UNAUTH\_UPDATE |  |  |  |  |  |  |  |  | X |  |  |  |
| T.HW\_ATTACK | X |  | X |  |  |  |  |  |  | X |  |  |
| A.CREDENTIAL\_REVOCATION |  |  |  |  |  |  |  |  |  | X |  |  |
| A.AUTH\_USERS |  |  |  |  |  |  |  |  |  |  | X |  |
| A.TRUSTED\_PEER |  |  |  |  |  |  |  |  |  |  |  | X |
| A.ROT\_INTEGRITY |  |  |  |  |  | X |  |  |  |  |  | X |

Table 2: Security Problem Definition Mapping to Security Objectives

The objectives can map to multiple assumptions or threats to fully define the objectives of the TOE and the operational environment.

# 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.

## SFR Architecture

A DSC implements all seven services in Table 3.

|  |  |
| --- | --- |
| Parse | FCS\_CKM.1 Cryptographic Key Generation  FCS\_CKM.2 Cryptographic Key Establishment  FCS\_COP.1/Hash Cryptographic Operation (Hashing)  FCS\_COP.1/HMAC Cryptographic Operation (Keyed Hash)  FCS\_COP.1/KAT Cryptographic Operation (Key Agreement/Transport)  FCS\_COP.1/KeyEnc Cryptographic Operation (Key Encryption)  FCS\_COP.1/PBKDF Cryptographic Operation (Password-Based Key Derivation Functions)  FCS\_COP.1/SKC Cryptographic Operation (Symmetric Key Cryptography)  FDP\_ACC.1 Subset Access Control  FDP\_ACF.1 Security Attribute Based Access Control  FDP\_IFC.1 Subset Information Flow Control  FDP\_IFF.1 Simple Security Attributes  FDP\_ITC.1 Import of User Data without Security Attributes  FDP\_ITC\_EXT.1 Parsing of SDEs  FDP\_ITC.2 Import of User Data with Security Attributes  FDP\_ITC\_EXT.2 Parsing of SDOs  FTP\_ITP\_EXT.1 Physically Protected Channel  FTP\_ITC\_EXT.1 Cryptographically Protected Communications Channels  FTP\_CCMP\_EXT.1 CCM Protocol  FTP\_GCMP\_EXT.1 GCM Protocol  FTP\_ITE\_EXT.1 Encrypted Data Communications |
| Provision | FCS\_CKM.1/AK Cryptographic Key Generation (Asymmetric Keys)  FCS\_CKM.1/KEK Cryptographic Key Generation Key Encryption Key (KEK)  FCS\_CKM\_EXT.5 Cryptographic Key Derivation  FCS\_COP.1/Hash Cryptographic Operation (Hashing)  FCS\_COP.1/HMAC Cryptographic Operation (Keyed Hash)  FCS\_COP.1/SKC Cryptographic Operation (Symmetric Key Cryptography)  FCS\_RBG\_EXT.1 Random Bit Generation  FDP\_ACC.1 Subset Access Control  FDP\_ACF.1 Security Attribute Based Access Control  FDP\_IFC.1 Subset Information Flow Control  FIA\_SOS.2 TSF Generation of Secrets  FMT\_MSA.3/SDO Static Attribute Initialization (Secure Data Objects)  FPT\_STM.1 Reliable Time Stamps  FCS\_ENT\_EXT.1 Entropy for External IT Entities  FCS\_RBG\_EXT.2 Cryptographic Operation Random Bit Generation  FCS\_CKM.1/SK Cryptographic Key Generation (Symmetric Key Encryption) |
| Protect | FCS\_COP.1/Hash Cryptographic Operation (Hashing)  FCS\_COP.1/HMAC Cryptographic Operation (Keyed Hash)  FCS\_COP.1/SKC Cryptographic Operation (Symmetric Key Cryptography)  FCS\_STG\_EXT.1 Secure Key Storage  FCS\_STG\_EXT.2 Key Storage Encryption  FCS\_STG\_EXT.3 Key Integrity Protection  FDP\_APW\_EXT.1 Storage of Authentication Tokens  FDP\_SDC\_EXT.1 Confidentiality of SDEs  FDP\_SDI.2 Stored Data Integrity Monitoring and Action  FIA\_USB\_EXT.1 Response to User Subject Binding  FMT\_SMR.2 Separation of Roles  FPT\_APW\_EXT.1 Protection of Administrator Passwords  FPT\_FLS.1/FI Failure with Preservation of Secure State (Fault Injection)  FPT\_MOD\_EXT.1 Debug Modes  FPT\_PHP.3 Resistance to Physical Attack  FPT\_ROT\_EXT.1 Root of Trust Services  FPT\_ROT\_EXT.2 Root of Trust for Storage  FPT\_PRO\_EXT.2 Data Integrity Measurements  FDP\_FRS\_EXT.2 Factory Reset Behavior  FIA\_AFL\_EXT.2 Authorization Failure Response  FPT\_FLS.1/FW Failure with Preservation of Secure State (Firmware)  FPT\_ITT.1 Basic Internal TSF Data Transfer Protection |
| Process | FCS\_COP.1/Hash Cryptographic Operation (Hashing)  FCS\_COP.1/HMAC Cryptographic Operation (Keyed Hash)  FCS\_COP.1/KeyEnc Cryptographic Operation (Key Encryption)  FCS\_COP.1/SigGen Cryptographic Operation (Signature Generation)  FCS\_COP.1/SigVer Cryptographic Operation (Signature Verification)  FCS\_COP.1/SKC Cryptographic Operation (Symmetric Key Cryptography)  FCS\_SLT\_EXT.1 Cryptographic Salt Generation  FDP\_ACC.1 Subset Access Control  FDP\_ACF.1 Security Attribute Based Access Control  FDP\_IFC.1 Subset Information Flow Control  FIA\_AFL\_EXT.1 Authorization Failure Handling  FIA\_SOS.2 TSF Generation of Secrets  FIA\_UIA\_EXT.1 User Identification and Authentication before any Action  FIA\_UAU.6/SDO Re-Authenticating (Access to SDO)  FIA\_USB\_EXT.1 Response to User Subject Binding  FMT\_MOF\_EXT.1 Management of Security Functions Behavior  FMT\_MSA.1/SDO Management of Security Attributes (Secure Data Objects)  FMT\_SMF.1 Specification of Management Functions  FMT\_SMR.2 Separation of Roles  FPT\_ROT\_EXT.1 Root of Trust Services  FPT\_RPL\_EXT.1 Replay Prevention  FPT\_STM.1 Reliable Time Stamps  FIA\_AFL\_EXT.2 Authorization Failure Response  FIA\_UAU.5/Prove Multiple Authentication Mechanisms (Prove Service) |
| Prove | FCS\_COP.1/Hash Cryptographic Operation (Hashing)  FCS\_COP.1/HMAC Cryptographic Operation (Keyed Hash)  FCS\_RBG\_EXT.1 Random Bit Generation  FCS\_SLT\_EXT.1 Cryptographic Salt Generation  FDP\_ACC.1 Subset Access Control  FDP\_ACF.1 Security Attribute Based Access Control  FPT\_PRO\_EXT.1 Root of Trust  FPT\_RPL\_EXT.1 Replay Prevention  FPT\_STM.1 Reliable Time Stamps  FCS\_RBG\_EXT.2 Cryptographic Operation Random Bit Generation  FPT\_ROT\_EXT.5 Root of Trust Reporting Mechanisms  FDP\_DAU.1/Prove Data Authentication for Use with the Prove Services  FDP\_MFW\_EXT.1 Mutable/Immutable Firmware  FDP\_MFW\_EXT.2 Basic Firmware Integrity  FDP\_MFW\_EXT.3 Firmware Authentication with Identity of Guarantor |
| Propagate | FCS\_COP.1/Hash Cryptographic Operation (Hashing)  FCS\_COP.1/HMAC Cryptographic Operation (Keyed Hash)  FCS\_COP.1/KeyEnc Cryptographic Operation (Key Encryption)  FCS\_COP.1/SKC Cryptographic Operation (Symmetric Key Cryptography)  FCS\_RBG\_EXT.1 Random Bit Generation  FCS\_SLT\_EXT.1 Cryptographic Salt Generation  FDP\_ACC.1 Subset Access Control  FDP\_ACF.1 Security Attribute Based Access Control  FDP\_ETC\_EXT.1 Secure Propagation of User Data  FDP\_IFC.1 Subset Information Flow Control  FDP\_IFF.1 Simple Security Attributes  FCS\_RBG\_EXT.2 Cryptographic Operation Random Bit Generation  FPT\_ITT.1 Basic Internal TSF Data Transfer Protection  FTP\_ITP\_EXT.1 Physically Protected Channel  FTP\_ITC\_EXT.1 Cryptographically Protected Communications Channels  FTP\_CCMP\_EXT.1 CCM Protocol  FTP\_GCMP\_EXT.1 GCM Protocol  FTP\_ITE\_EXT.1 Encrypted Data Communications |
| Purge | FCS\_CKM.4 Cryptographic Key Destruction  FCS\_CKM\_EXT.4 Cryptographic Key and Key Material Destruction Timing  FCS\_RBG\_EXT.1 Random Bit Generation  FDP\_RIP.1 Subset Residual Information Protection  FCS\_RBG\_EXT.2 Cryptographic Operation Random Bit Generation  FDP\_FRS\_EXT.2 Factory Reset Behavior |
| TSF Security | FDP\_FRS\_EXT.1 Factory Reset  FDP\_MFW\_EXT.1 Mutable/Immutable Firmware  FMT\_SMF.1 Specification of Management Functions  FPT\_FLS.1/FI Failure with Preservation of Secure State (Fault Injection)  FPT\_MOD\_EXT.1 Debug Modes  FPT\_PHP.3 Resistance to Physical Attack  FPT\_TST.1 Integrity Checking  FRU\_FLT.1 Degraded Fault Tolerance  FPT\_PRO\_EXT.2 Data Integrity Measurements  FDP\_MFW\_EXT.2 Basic Firmware Integrity  FDP\_MFW\_EXT.3 Firmware Authentication with Identity of Guarantor  FDP\_FRS\_EXT.2 Factory Reset Behavior  FPT\_FLS.1/FW Failure with Preservation of Secure State (Firmware)  FPT\_RPL.1/Rollback Replay Detection (Rollback) |

Table 3: SFR Architecture

## Conventions

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

* Unaltered SFRs are stated in the form used in [CC2] or their extended component definition (ECD);
* Refinement made in the PP: the added/removed text is indicated with **bold text**/~~strikethroughs~~. When text is substituted (i.e. some text is added in place of some other text, which is then deleted), only the added text is included;

Note that a refinement is also used to indicate cases where the PP replaces an assignment defined for an SFR in [CC2] and replaces it with a selection;

* Selections:
  + wholly or partially completed in the PP: the selection values (i.e. the selection values adopted in the PP or the remaining selection values available for the ST) are indicated with underlined text;

e.g. “[*selection: disclosure, modification, loss of use*]” in [CC2] or an ECD might become “disclosure” (completion) or “[selection: disclosure, modification]” (partial completion) in the PP;

* + Some SFRs include selections that determine or constrain other assignments or selections. In these cases, a table follows the requirement in which each row of the table defines a permitted set of choices. Each row includes a unique identifier defined solely to provide a label for the selection set. Individual entries in these tables may also require further selections or assignments.

e.g. for FCS\_CKM.1/AK (see Table 4), the ST for a TOE that supports RSA keys must include the entries for ‘key name’, ‘key sizes’, and ‘list of standards’ as specified in row 1AK. For ‘key sizes’, the ST author must further select which of the required key sizes are supported. The row identifiers are merely intended as quick-reference handles—there is no expectation that the TSF actually refer internally to RSA keys using this identifier. Likewise, if the TOE supports ECC the ST must include the entries from row 2AK along with the appropriate selections.

| Identifier | Key Name | Key Sizes | List of Standards |
| --- | --- | --- | --- |
| 1AK | RSA | [selection: 2048 bit, 3072 bit] | FIPS PUB 186-4 (Section B.3) |
| 2AK | ECC | [selection: 256 (P-256), 384 (P-384), 512 (P-521)] | FIPS PUB 186-4 (Section B.4 & D.1.2) |
| 3AK | BPC | [selection: 256 (brainpoolP256r1), 384 (brainpoolP384r1), 512 (brainpoolP512r1)] | RFC5639 (Section 3) [Brainpool Curves]  FIPS PUB 186-4 (Section B.4) |

Table 4: Sample Cryptographic Table

* Assignment wholly or partially completed in the PP: indicated with *italicized text*;
* Assignment completed within a selection in the PP: the completed assignment text is indicated with *italicized and underlined text*

e.g. “[selection: change\_default, query, modify, delete, [*assignment: other operations*]]” in [CC2] or an ECD might become “[change\_default, [*select\_tag*]]” (completion of both selection and assignment) or “[selection: change\_default, select\_tag, [*select\_value*]]” (partial completion of selection, and completion of assignment) in the PP;

* Iteration: indicated by adding a string starting with “/” (e.g. "FCS\_COP.1/Hash”).

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 are 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 [CC2] are identified by having a label ‘\_EXT’ at the end of the SFR name.

## Cryptographic Support

### FCS\_CKM.1 Cryptographic Key Generation

**FCS\_CKM.1 Cryptographic Key Generation**

**FCS\_CKM.1.1** The TSF shall generate cryptographic keys **by [selection: parsing in accordance with [selection: FDP\_ITC.1, FDP\_ITC.2], asymmetric key generation in accordance with FCS\_CKM.1/AK, symmetric key generation in accordance to FCS\_CKM.1/SK]** ~~in accordance with a specified cryptographic key generation algorithm [assignment:~~ *~~cryptographic key generation algorithm~~*~~] and specified cryptographic key sizes [assignment:~~ *~~cryptographic key sizes~~*~~] that meet the following: [assignment:~~ *~~list of standards~~*~~]~~.

This SFR allows the ST to support both the importation of keys from external sources (FDP\_ITC.1, FDP\_ITC.2) as well as internal generation of keys (FCS\_CKM.1/AK and FCS\_CKM.1/SK).

### FCS\_CKM.1/AK Cryptographic Key Generation (Asymmetric Keys)

**FCS\_CKM.1/AK Cryptographic Key Generation (Asymmetric Keys)**

**FCS\_CKM.1.1/AK** The TSF shall generate **asymmetric** cryptographic **keys [selection: key name from Table 5 below] with** specified cryptographic key sizes **[selection: key size from Table 5 below]** that meet the following: **[selection: list of standards from Table 5 below].**

| Identifier | Key Name | Key Sizes | List of Standards |
| --- | --- | --- | --- |
| AK1 | RSA | [selection: 2048 bit, 3072 bit] | FIPS PUB 186-4 (Section B.3) |
| AK2 | ECC-N | [selection: 256 (P-256), 384 (P-384), 512 (P-521)] | FIPS PUB 186-4 (Section B.4 & D.1.2) |
| AK3 | ECC-B | [selection: 256 (brainpoolP256r1), 384 (brainpoolP384r1), 512 (brainpoolP512r1)] | RFC5639 (Section 3) [Brainpool Curves]  FIPS PUB 186-4 (Section B.4) |

Table 5: Supported Methods for Asymmetric Key Generation

This requirement is included for the purposes of encryption and decryption operations only. To support ITE protected communications requirement for the transfer of encrypted data, this requirement mandates implementation compliance to FIPS 186-4 only. Implementations according to FIPS 186-2 or FIPS 186-3 will not be accepted.

### FCS\_CKM.1/KEK Cryptographic Key Generation (Key Encryption Key)

**FCS\_CKM.1/KEK Cryptographic Key Generation (Key Encryption Key)**

**FCS\_CKM.1.1/KEK** The TSF shall generate **key encryption keys** in accordance with a specified cryptographic key generation algorithm **corresponding to** **[selection:**

* **Asymmetric KEKs generated in accordance with FCS\_CKM.1/AK,**
* **Symmetric KEKs generated in accordance with FCS\_CKM.1/SK,**
* **Derived KEKs generated in accordance with FCS\_CKM\_EXT.5**

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

KEKs protect KEKs and Symmetric Keys (SKs). DSCs should use key strengths commensurate with protecting the chosen symmetric encryption key strengths.

### FCS\_CKM.2 Cryptographic Key Establishment

**FCS\_CKM.2 Cryptographic Key Establishment**

**FCS\_CKM.2.1** The TSF shall **establish** cryptographic **keys** in accordance with a specified cryptographic key distribution method: **[selection:**

* **RSA-based key establishment schemes that meet the following: NIST Special Publication 800-56B Revision 1, “Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography”;**
* **Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 3, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”;**
* **FFC-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 3, “recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”;**
* **Key establishment scheme using Diffie-Hellman group 14 that meets the following: RFC 3526, Section 3**

**]** ~~that meets the following: [~~*~~assignment: list of standards~~*~~]~~.

This is a refinement of the SFR FCS\_CKM.2 to deal with key establishment rather than key distribution.

The ST author selects all key establishment schemes used for the selected cryptographic protocols.

The RSA-based key establishment schemes are described in Section 9 of NIST SP 800-56B Revision 1 [NIST-RSA]; however, Section 9 relies on implementation of other sections in SP 800-56B Revision 1.

The elliptic curves used for the key establishment scheme correlate with the curves specified in FCS\_CKM.1/AK.

The selections in this SFR must be consistent with those for FCS\_COP.1/KAT.

### FCS\_CKM.4 Cryptographic Key Destruction

**FCS\_CKM.4 Cryptographic Key Destruction**

**FCS\_CKM.4.1** The TSF shall destroy cryptographic keys **and keying material** in accordance with a specified cryptographic key destruction **method [selection:**

1. **For volatile memory, the destruction shall be executed by a [selection:** 
   1. **single overwrite consisting of [selection:** 
      1. **a pseudo-random pattern using the TSF’s RBG,**
      2. **zeroes,**
      3. **ones,**
      4. **a new value of a key,**
      5. **[*assignment: some value that does not contain any CSP*]],**
   2. **removal of power to the memory,**
   3. **removal of all references to the key directly followed by a request for garbage collection];**
2. **For non-volatile memory [selection:**
   1. **that employs a wear-leveling algorithm, the destruction shall be executed by a [selection:** 
      1. **single overwrite consisting of [selection: zeroes, ones, pseudo-random pattern, a new value of a key of the same size, [*assignment: some value that does not contain any CSP*]],**
      2. **block erase];**
   2. **that does not employ a wear-leveling algorithm, the destruction shall be executed by a [selection:** 
      1. **[selection: single, [*assignment: ST author defined multi-pass*]] overwrite consisting of [selection: zeros, ones, pseudo-random pattern, a new value of a key of the same size, [*assignment: some value that does not contain any CSP*]] followed by a read-verify. If the read-verification of the overwritten data fails, the process shall be repeated again up to [*assignment: number of times to attempt overwrite*] times, whereupon an error is returned.**
      2. **block erase]**

**]**

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

*A DSC must implement mechanisms to destroy cryptographic keys and key material contained in persistent storage when no longer needed. The term “cryptographic keys” in this SFR includes the authorization data that is the entry point to a key chain and all other cryptographic keys and keying material (whether in plaintext or encrypted form). This SFR does not apply to the public component of asymmetric key pairs, or to keys that are permitted to remain stored such as device identification keys.*

In the case of volatile memory, the selection “destruction of reference to the key directly followed by a request for garbage collection” is used in a situation where the TSF cannot address the specific physical memory locations holding the data to be erased and therefore relies on addressing logical addresses (which frees the relevant physical addresses holding the old data) and then requesting the platform to ensure that the data in the physical addresses is no longer available for reading (i.e. the “garbage collection” referred to in the SFR text).

Guidance documentation for the TOE requires users not to allow the TOE to leave the user’s control while a session is active (and hence while the DEK is likely to be in plaintext in volatile memory).

The selection for destruction of data in non-volatile memory includes block erase as an option, and this option applies only to flash memory. A block erase does not require a read verify, since the mappings of logical addresses to the erased memory locations are erased as well as the data itself.

Where different destruction methods are used for different data and/or different destruction situations then the different methods and the data/situations they apply to (e.g. different points in time, or power-loss situations) are described in the TSS (and the ST may use separate iterations of the SFR to aid clarity). The TSS includes a table describing all relevant keys and keying material (including authorization data) used in the implementation of the SFRs, stating the source of the data, all memory types in which the data is stored (covering storage both during and outside of a session, and both plaintext and non-plaintext forms of the data), and the applicable destruction method and time of destruction in each case.

*Some selections allow assignment of “a value that does not contain any sensitive data”. This means that the TOE uses some specified data not drawn from an RBG meeting FCS\_RBG\_EXT requirements, and not being any of the particular values listed as other selection options. The point of the phrase “does not contain any sensitive data” is to ensure that the overwritten data is carefully selected, and not taken from a general pool that might contain current or residual data (e.g. user data or intermediate key chain values) that itself requires confidentiality protection.*

### FCS\_CKM\_EXT.4 Cryptographic Key and Key Material Destruction Timing

**FCS\_CKM\_EXT.4 Cryptographic Key and Key Material Destruction**

**FCS\_CKM\_EXT.4.1** The TSF shall destroy all keys and keying material when no longer needed.

The DSC will have mechanisms to destroy keys, including intermediate keys and key material, by using an approved method, FCS\_CKM.4. Examples of keys include intermediate keys, leaf keys, encryption keys, signing keys, verification keys, authentication tokens, and submasks. The DSC will have mechanisms to destroy keys and key material contained in persistent storage when no longer needed. Based on their implementation, vendors will explain when certain keys are no longer needed. An example in which key is no longer necessary includes a wrapped key whose password has changed. However, there are instances when keys are allowed to remain in memory, for example, a device identification key.

### FCS\_CKM\_EXT.5 Cryptographic Key Derivation

**FCS\_CKM\_EXT.5 Cryptographic Key Derivation**

**FCS\_CKM\_EXT.5.1** The TSF shall generate cryptographic keys **[selection: key name(s) as chosen from Table 6 below]** with specified cryptographic key sizes **[selection: key size(s) as chosen from Table 6 below for corresponding key type(s)]** that meet the following: **[selection: list of standard(s) as chosen from Table 6 below for corresponding key name(s)]**.

| Identifier | Key Type | Input Parameters | Key Derivation Algorithm | Key Sizes | List of Standards |
| --- | --- | --- | --- | --- | --- |
| KeyDrv1 | [*assignment: key name*] | Direct Generation froma Random Bit Generator as specified in FCS\_RBG\_EXT.1 | KDF in Counter Mode using [selection:  CMAC-AES-128; CMAC-AES-192; CMAC-AES-256; HMAC-SHA-1; HMAC-SHA-256; HMAC-SHA-512] as the PRF | [selection: 128, 256 bits] | NIST SP 800-108 (Section 5.1) [KDF in Counter Mode]  [selection:  ISO-CMAC; NIST-CMAC; ISO-CIPH; ISO-HMAC; FIPS-HMAC; ISO-HASH; FIPS-SHA] |
| KeyDrv2 | [*assignment: key name*] | Direct Generation froma Random Bit Generator as specified in FCS\_RBG\_EXT.1 | KDF in Feedback Mode using [selection:  CMAC-AES-128; CMAC-AES-192; CMAC-AES-256; HMAC-SHA-1; HMAC-SHA-256; HMAC-SHA-512] as the PRF | [selection: 128, 256 bits] | NIST SP 800-108 (Section 5.2) [KDF in Feedback Mode]  [selection:  ISO-CMAC; NIST-CMAC; ISO-CIPH;  ISO-HMAC; FIPS-HMAC; ISO-HASH; FIPS-SHA] |
| KeyDrv3 | [*assignment: key name*] | Direct Generation froma Random Bit Generator as specified in FCS\_RBG\_EXT.1 | KDF in Double-Pipeline Iteration Mode using [selection:  CMAC-AES-128; CMAC-AES-192; CMAC-AES-256, HMAC-SHA-1; HMAC-SHA-256; HMAC-SHA-512] as the PRF | [selection: 128, 256 bits] | NIST SP 800-108 (Section 5.3) [KDF in Double-Pipeline Iteration Mode]  [selection: ISO-CMAC; NIST-CMAC, ISO-CIPH; ISO-HMAC; FIPS-HMAC; ISO-HASH; FIPS-SHA] |
| KeyDrv4 | [*assignment: key name*] | Intermediary keys | [selection: exclusive OR (XOR); SHA-256; SHA-512] | [selection: 128, 256 bits] | [selection:  ISO-HASH; FIPS-SHA] |
| KeyDrv5 | [*assignment: key name*] | Concatenated keys | [selection: KeyDrv1, KeyDrv2, KeyDrv3] from this table | [selection: 128, 256] bits | [selection: see List of Standards for corresponding algorithm] from this table. |
| KeyDrv6 | [*assignment: key name*] | Two keys | [selection: AES-CCM, AES-GCM, AES-CBC, AES-KWP, AES-KW, CAM-CBC, CAM-CCM, CAM-GCM] from FCS\_COP.1/SKC Symmetric Key table | [selection: see Key Sizes in FCS\_COP.1/SKC Symmetric Key table] | [selection: see List of Standards in FCS\_COP.1/SKC Symmetric Key table] |
| KeyDrv7 | [selection: symmetric key, secret IV, seed] | Shared secret, salt, output length, fixed information | [selection: hash function from FCS\_COP.1/Hash, keyed hash from FCS\_COP.1/HMAC] | [selection: 128, 256] bits | [NIST-KDRV] sec 4  [selection: see List of Standards in FCS\_COP.1/Hash and FCS\_COP.1/HMAC] |
| KeyDrv8 | [selection: symmetric key, secret IV, seed] | Shared secret, salt, IV, output length, fixed information | [selection: keyed hash from FCS\_COP.1/HMAC] | [selection: 128, 256] | [NIST-KDRV] sec 5  [selection: see List of Standards in FCS\_COP.1/HMAC] |

Table 6: Key Derivation Functions

*The interface referenced in the requirement could take different forms, the most likely of which is an application programming interface to an OS kernel. There may be various levels of abstraction*

For Authorization Factor Submasks, the key size to be used in the HMAC falls into a range between L1 and L2 defined in ISO/IEC 10118 for the appropriate hash function (for example for SHA-256 L1 = 512, L2 =256) where L2 ≤ k ≤ L1.

General note: in order to use a NIST SP 800-108 conformant method of key derivation, the TOE is permitted to implement this with keys as derived as indicated in Key Derivation Functions table above, and with the algorithms as indicated in the same table.

NIST SP 800-131A Rev 1 allows the use of SHA-1 in these use cases.

KeyDrv5, KeyDrv6, and the XOR option in KeyDrv4 will create an “inverted key hierarchy” in which the TSF will combine two or more keys to create a third key. These same KDFs may also use a submask key as input, which could be an authorization factor or derived from a PBKDF. In these cases the ST author must explicitly declare this option and should present a reasonable argument that the entropy of the inputs to the KDFs will result in full entropy of the expected output.

If keys are combined, the ST author shall describe which method of combination is used in order to justify that the effective entropy of each factor is preserved.

The documentation of the product's encryption key management should be detailed enough that, after reading, the evaluator will thoroughly understand the product's key management and how it meets the requirements to ensure the keys are adequately protected. This documentation should include an essay and diagram(s). This documentation is not required to be part of the [TSS](https://www.niap-ccevs.org/MMO/PP/-417-/#abbr_TSS) - it can be submitted as a separate document and marked as developer proprietary.

SP 800-56C specifies a two-step key derivation procedure that employs an extraction-then-expansion technique for deriving keying material from a shared secret generated during a key establishment scheme. The Randomness Extraction step as described in Section 5 of SP 800-56C is followed by Key Expansion using the key derivation functions defined in SP 800-108 (as described in Section 6 of SP 800-56C).

### FCS\_COP.1/Hash Cryptographic Operation (Hashing)

**FCS\_COP.1/Hash Cryptographic Operation (Hashing)**

**FCS\_COP.1.1/Hash** The TSF shall perform [*cryptographic hashing*] in accordance with a specified cryptographic algorithm **[selection: SHA-1, SHA-256, SHA-384, SHA-512, SHA-3-224, SHA-3-256, SHA-3-384, SHA-3-512]** that meets the following: **[selection: ISO/IEC 10118-3:2018, FIPS 180-4]**.

The hash selection should be consistent with the overall strength of the algorithm used for signature generation. For example, the DSC should choose SHA-256 for 2048-bit RSA or ECC with P-256, SHA-384 for 3072-bit RSA, 4096-bit RSA, or ECC with P-384, and SHA-512 for ECC with P-521. The DSC selects the standard based on the algorithms selected.

SHA-1 may be used for the following applications: generating and verifying hash-based message authentication codes (HMACs), key derivation functions (KDFs), and random bit/number generation[[2]](#footnote-3).

An upcoming version of the ISO/IEC 10118-3 standard is being drafted as of the publication of this cPP. It will be added as a possible selection as soon as it is published.

### FCS\_COP.1/HMAC Cryptographic Operation (Keyed Hash)

**FCS\_COP.1/HMAC Cryptographic Operation (Keyed Hash)**

**FCS\_COP.1.1/HMAC** The TSF shall perform [*keyed hash message authentication*] in accordance with a specified cryptographic algorithm **[selection: HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512, KMAC128, KMAC256]** and cryptographic key sizes **[selection: key size (in bits) used in HMAC]** that meet the following: [*ISO/IEC 9797-2:2011, Section 7 “MAC Algorithm 2”, [NIST-KDV3] section 4 “KMAC”*].

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

Note that HMAC-SHA-1 is permissible because it can currently be used for wireless LAN communications. It is anticipated that this will be removed from a future version of this cPP.

### FCS\_COP.1/KAT Cryptographic Operation (Key Agreement/Transport)

**FCS\_COP.1/KAT Cryptographic Operations (Key Agreement/Transport)**

**FCS\_COP.1.1/KAT** The TSF shall perform [*cryptographic key agreement/transport*] in accordance with a specified cryptographic algorithm **[selection: cryptographic algorithm from Table 7 below]** and cryptographic key sizes **[selection: key sizes from Table 7 below]** that meet the following: **[selection: list of standards from Table 7 below]**.

| Identifier | Cryptographic Algorithm | Key Sizes | List of Standards |
| --- | --- | --- | --- |
| KAS1 | RSA-single party | [selection: 2048 bit, 3072 bit] | NIST SP 800-56Br2 section 8.2 |
| KAS2 | RSA-both party | [selection: 2048 bit, 3072 bit] | NIST SP 800-56Br2 section 8.3 |
| KTS-OAEP | RSA | [selection: 2048 bit, 3072 bit] | NIST SP 800-56Br2 section 9 |
| KTS-KEM-KWS | RSA | [selection: 2048 bit, 3072 bit] | NIST SP 800-56Br1 |
| ECDH-NIST | ECDH with NIST curves | [selection: 256 (P-256), 384 (P-384), 512 (P-521)] | NIST SP 800-56A |
| ECDH-BPC | ECDH with Brainpool curves | [selection: 256 (brainpoolP256r1), 384 (brainpoolP384r1), 512 (brainpoolP512r1)] | NIST SP 800-56A |
| DHG14 | Diffie-Hellman Group 14 | [selection: 2048 bit, 3072 bit] | RFC 3526, Section 3 |

Table 7: Supported Methods for Key Agreement/Transport Operation

*The selections in this SFR should be consistent with the algorithms selected in FCS\_CKM.2.*

### FCS\_COP.1/KeyEnc Cryptographic Operation (Key Encryption)

**FCS\_COP.1/KeyEnc Cryptographic Operation (Key Encryption)**

**FCS\_COP.1.1/KeyEnc** The TSF shall perform [*key encryption and decryption*] in accordance with a specified cryptographic algorithm **[selection: cryptographic algorithm from Table 8 below]** and the cryptographic key size **[selection: key size from Table 8 below]** that meet the following: **[selection: list of standards from Table 8 below]**.

| Identifier | Cryptographic Algorithm | Key Size | List of Standards |
| --- | --- | --- | --- |
| SE1 | Symmetric [selection: AES-CCM, AES-GCM, AES-CBC, AES-KWP, AES-KW] | [selection: 128, 192, 256] bits | See FCS\_COP.1/SKC |
| SE2 | Symmetric [selection AES-KWP, AES-KW, CAM-CBC, CAM-CCM, CAM-GCM] | [selection: 128, 256] bits | See FCS\_COP.1/SKC |
| AE1 | Asymmetric [selection: KTS-OAEP, KTS-KEM-KWS] | [selection: 2048, 3027] bits | See FCS\_COP.1/KAT |
| XOR | Exclusive OR operation | [selection: 128, 192, 256] bits | See FCS\_CKM\_EXT.5 |

Table 8: Supported Methods for Key Encryption Operation

A TOE will use this requirement to specify how the Key Encryption Key (KEK) wraps a symmetric encryption key. A TOE will always need this requirement in order to capture the last stage of the key chain in which the Key Encryption Key (KEK) wraps the symmetric encryption key.

### FCS\_COP.1/PBKDF Cryptographic Operation (Password-Based Key Derivation Functions)

**FCS\_COP.1/PBKDF Cryptographic Operation (Password-Based Key Derivation Functions)**

**FCS\_COP.1.1/PBKDF** The TSF shall perform [*password-based key derivation functions*] in accordance with a specified cryptographic algorithm [*HMAC-[selection: SHA-256, SHA-384, SHA-512]*], with **[*assignment: integer number greater than or equal to 1000*] iterations, and output cryptographic key sizes [selection: 128, 256] bits** that meet the following standard: [*NIST SP 800-132*].

*This password must be conditioned into a string of bits that forms the submask to be used as input into the KEK. Conditioning can be performed using one of the identified hash functions or the process described in NIST SP 800-132; the method used is selected by the ST Author. NIST SP 800-132 requires the use of a pseudo-random function (PRF) consisting of HMAC with an approved hash function. The ST author selects the hash function used, also includes the appropriate requirements for HMAC and the hash function.*

*Appendix A of NIST SP 800-132 recommends setting the iteration count in order to increase the computation needed to derive a key from a password and, therefore, increase the workload of performing a dictionary attack.*

### FCS\_COP.1/SigGen Cryptographic Operation (Signature Generation)

**FCS\_COP.1/SigGen Cryptographic Operation (Signature Generation)**

**FCS\_COP.1.1/SigGen** The TSF shall perform [*digital signature generation*] in accordance with a specified cryptographic algorithm [**selection: cryptographic algorithm from Table 9 below**] and cryptographic key sizes [**selection: key sizes from Table 9 below**] that meet the following: [**selection: list of standards from Table 9 below**].

| Identifier | Cryptographic Algorithm | Key sizes | List of Standards |
| --- | --- | --- | --- |
| SigGen1 | RSASSA-PKCS1-v1\_5 using [selection: SHA-256, SHA-512] | [selection: 2048 bit, 3072 bit] | [selection: RFC3447, PKCS #1 v2.1 (Section 8.2), FIPS186-4, (Section 5.5)] [RSASSA-PKCS1-v1\_5]  [selection: ISO10118-3, (Section 10, 11); FIPS180-4, (Section 6)] [SHA] |
| SigGen2 | Digital signature scheme 2 using [selection: SHA-256, SHA-512] | [selection: 2048 bit, 3072 bit] | ISO9796-2, (Section 9) [Digital signature scheme 2]  [selection: ISO10118-3, (Section 10, 11); FIPS180-4, (Section 6)] [SHA] |
| SigGen3 | Digital signature scheme 3 using [selection: SHA-256, SHA-512] | [selection: 2048 bit, 3072 bit] | ISO9796-2, (Section 10) [Digital signature scheme 3]  [selection: ISO10118-3, (Section 10, 11); FIPS180-4, (Section 6)] [SHA] |
| SigGen4 | RSASSA-PSS using [selection: SHA-256, SHA-512] | [selection: 2048 bit, 3072 bit] | RFC3447, PKCS#1v2.1 (Section 8.1) [RSASSA-PSS]  [selection: ISO10118-3, (Section 10, 11); FIPS180-4, (Section 6)] [SHA] |
| SigGen5 | ECDSA on [selection: brainpoolP256r1, brainpoolP384r1, brainpoolP512r1, NIST P-256, NIST P-384, NIST P-521] using [selection: SHA-256, SHA-512] | [selection: 256 bits, 384 bits, 512 bits] | [selection: ISO14888-3; FIPS186-4 (Section 6)] [ECDSA]  RFC5639 (Section 3) [Brainpool Curves]  FIPS186-4 (Appendix D.1.2) [NIST Curves]  [selection: ISO10118-3, (Section 10, 11); FIPS180-4, (Section 6)] [SHA] |

Table 9: Supported Methods for Signature Generation Operation

### FCS\_COP.1/SigVer Cryptographic Operation (Signature Verification)

**FCS\_COP.1/SigVer Cryptographic Operation (Signature Verification)**

**FCS\_COP.1.1/SigVer** The TSF shall perform [*digital signature verification for authenticity*] in accordance with a specified cryptographic algorithm [**selection: cryptographic algorithm from Table 10 below**] and cryptographic key sizes [**selection: key size from Table 10 below**] that meet the following: [**selection: list of standards from Table 10 below**].

| Identifier | Cryptographic Algorithm | Key Sizes | List of Standards |
| --- | --- | --- | --- |
| SigVer1 | RSASSA-PKCS1-v1\_5 using [selection: SHA-256, SHA-512] | [selection: 2048 bit, 3072 bit] | [selection: RFC3447, PKCS #1 v2.1 (Section 8.2), FIPS186-4, (Section 5.5)] [RSASSA-PKCS1-v1\_5]  [selection: ISO10118-3, (Section 10, 11); FIPS180-4, (Section 6)] [SHA] |
| SigVer2 | Digital signature scheme 2 using [selection: SHA-256, SHA-512] | [selection: 2048 bit, 3072 bit] | ISO9796-2, (Section 9) [Digital signature scheme 2]  [selection: ISO10118-3, (Section 10, 11); FIPS180-4, (Section 6)] [SHA] |
| SigVer3 | Digital signature scheme 3 using [selection: SHA-256, SHA-512] | [selection: 2048 bit, 3072 bit] | ISO9796-2, (Section 10) [Digital signature scheme 3]  [selection: ISO10118-3, (Section 10, 11); FIPS180-4, (Section 6)] [SHA] |
| SigVer4 | RSASSA-PSS using [selection: SHA-256, SHA-512] | [selection: 2048 bit, 3072 bit] | RFC3447, PKCS#1v2.1 (Section 8.1) [RSASSA-PSS]  [selection: ISO10118-3, (Section 10, 11); FIPS180-4, (Section 6)] [SHA] |
| SigVer5 | ECDSA on [selection: brainpoolP256r1, brainpoolP384r1, brainpoolP512r1, NIST P-256, NIST P-384, NIST P-521] using [selection: SHA-256, SHA-512] | [selection: 256 bits, 384 bits, 512 bits] | [selection: ISO14888-3; FIPS186-4 (Section 6)] [ECDSA]  RFC5639 (Section 3) [Brainpool Curves]  FIPS186-4 (Appendix D.1.2) [NIST Curves]  [selection: ISO10118-3, (Section 10, 11); FIPS180-4, (Section 6)] [SHA] |

Table 10: Supported Methods for Signature Verification Operation

### FCS\_COP.1/SKC Cryptographic Operation (Symmetric Key Cryptography)

**FCS\_COP.1/SKC Cryptographic Operation (Symmetric Key Cryptography)**

**FCS\_COP.1.1/SKC** The TSF shall perform [*TSF data encryption/decryption*] in accordance with a specified cryptographic algorithm **[selection: cryptographic algorithm from Table 11 below]** and cryptographic key sizes **[selection: cryptographic key sizes from Table 11 below]** that meet the following: **[selection: list of standards from Table 11 below]**.

| Identifier | Cryptographic Algorithm | Key Sizes | List of Standards |
| --- | --- | --- | --- |
| AES-CCM | AES in CCM mode with unpredictable, non- repeating nonce, minimum size of 64 bits | [selection: 128 bits, 192 bits, 256 bits] | ISO 18033-3 (AES)  ISO 19772, sec. 8 (CCM)  NIST SP800-38C (CCM) |
| AES-GCM | AES in GCM mode with non- repeating IVs IV length must be equal to 96 bits; the deterministic IV construction method [SP800- 38D, Section 8.2.1] must be used; the MAC length t must be one of the values 96, 104, 112, 120, and 128 bits. | [selection: 128 bits, 192 bits, 256 bits] | ISO 18033-3 (AES)  ISO 19772, sec.11 (GCM)  NIST SP800-38D (GCM) |
| AES-CBC | AES in CBC mode with non-repeating and unpredictable IVs | [selection: 128 bits, 192 bits, 256 bits] | ISO 18033-3 (AES) ISO 10116 (CBC)  NIST SP800-38A (CBC) |
| XTS-AES | AES in XTS mode with unique [selection: consecutive non- negative integers starting at an arbitrary non-negative integer, data unit sequence numbers] tweak values | [selection: 128 bits, 192 bits, 256 bits] | ISO 18033-3 (AES)  [selection: IEEE 1619, NIST SP800-38E] (XTS) |
| AES-KWP | KWP based on AES | [selection: 128 bits, 256 bits] | ISO/IEC 18033-3 (AES), NIST SP 800-38F, sec. 6.3 (KWP)  ISO/IEC 19772, clause 7 (key wrap) |
| AES-KW | KW based on AES | [selection: 128 bits, 256 bits] | ISO/IEC 18033-3 (AES), NIST SP 800-38F, sec. 6.2 (KW)  ISO/IEC 19772, clause 7 (key wrap) |
| CAM-CBC | Camellia in CBC mode with non-repeating and unpredictable IVs | [selection: 128 bits, 256 bits] | ISO 18033-3 (Camellia)  ISO 10116 (CBC) |
| CAM-CCM | Camellia in CCM mode with unpredictable, non-repeating nonce, minimum size of 64 bits | [selection: 128 bits, 256 bits] | ISO 18033-3 (Camellia)  ISO 19772, sec. 8 (CCM)  SP800-38C |
| CAM-GCM | Camellia in GCM mode with non-repeating IVs; the IV length must be equal to 96 bits; the deterministic IV construction method [SP800- 38D, Section 8.2.1] must be used; the MAC length t must be one of the values 96, 104, 112, 120, and 128 bits. | [selection: 128 bits, 256 bits] | ISO 18033-3 (Camellia)  ISO 19772, sec.11  (GCM)  NIST SP800-38D |
| XTS-CAM | Camellia in XTS mode with unique [selection: consecutive non-negative integers starting at an arbitrary non-negative integer, data unit sequence numbers] tweak values | [selection: 256 bits, 512 bits] | ISO 18033-3 (Camellia)  [selection: IEEE 1619,  SP800-38E] (XTS) |

Table 11: Supported Methods for Symmetric Key Cryptography Operation

### FCS\_RBG\_EXT.1 Random Bit Generation

**FCS\_RBG\_EXT.1 Random Bit Generation**

**FCS\_RBG\_EXT.1.1** The TSF shall perform all deterministic random bit generation services in accordance with ISO/IEC 18031:2011 using [selection: Hash\_DRBG (any), HMAC\_DRBG (any), CTR\_DRBG (AES)].

**FCS\_RBG\_EXT.1.2** The deterministic RBG shall be seeded by at least one entropy source in accordance with NIST SP 800-90B that accumulates entropy from [selection: [*assignment: number of software-based sources*] software-based noise source, [*assignment: number of hardware-based sources*] hardware-based noise source] with a minimum of [selection: 128 bits, 192 bits, 256 bits] of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011, of the keys and CSPs that it will generate.

NIST Special Pub 800-90B, Appendix C describes the minimum entropy measurement that products should use immediately, and that this cPP will require in the future. [NIST-DRBG] should be referenced for other applications of the DRBG until 800-90B is officially published.

ISO/IEC 18031:2011 contains four different methods of generating random numbers. Each of these in turn depends on underlying cryptographic primitives (hash functions/ciphers). This cPP allows SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512 for Hash\_DRBG or HMAC\_DRBG and only AES-based implementations for CTR\_DRBG.

### FCS\_SLT\_EXT.1 Cryptographic Salt Generation

**FCS\_SLT\_EXT.1 Salt Generation**

**FCS\_SLT\_EXT.1.1** The TSF shall use salts and nonces of bit-length greater than or equal to 128 bits generated by an RBG as specified in FCS\_RBG\_EXT.1.

### FCS\_STG\_EXT.1 Secure Key Storage

**FCS\_STG\_EXT.1 Secure Key Storage**

**FCS\_STG\_EXT.1.1** The TSF shall provide [selection: mutable hardware, software-based] secure key storage for asymmetric private keys and [selection: symmetric keys, persistent secrets, no other keys].

A hardware key store can be exposed to the TSF through a variety of interfaces, including embedded on the motherboard, USB, microSD, and Bluetooth.

Immutable hardware is considered outside of this requirement and will be covered elsewhere (see FMT\_MSA.3/SDO).

If the secure key storage is implemented in software that is protected as required by FCS\_STG\_EXT.2, the ST author is expected to select “software-based.” If “software-based” is selected, the ST author is expected to select “all software-based key storage” in FCS\_STG\_EXT.2.

Support for secure key storage for all symmetric keys and persistent secrets will be required in future revisions.

**FCS\_STG\_EXT.1.2** The TSF shall support the capability of importing keys/secrets into the TOE upon request of [selection: a user, an administrator**].**

**FCS\_STG\_EXT.1.3** The TSF shall be capable of destroying keys/secrets in the secure key storage upon request of [selection: the user, the administrator].

**FCS\_STG\_EXT.1.4** The TSF shall have the capability to allow only the application that imported the key/secret the use of the key/secret. Exceptions may only be explicitly authorized by [selection: the user, the administrator, a common application developer].

**FCS\_STG\_EXT.1.5** The TSF shall allow only the application that imported the key/secret to request that the key/secret be destroyed. Exceptions may only be explicitly authorized by [selection: the user, the administrator, a common application developer].

**FCS\_STG\_EXT.1.6** The TSF shall prevent recovery of a symmetric encryption key from persistent storage without proper authorization.

*If the product writes an SK to persistent storage, no one can recover that key from persistent storage unless they provide a proper authorization factor the product uses to recover the key. As an example, if the product XORs an SK with a constant and writes the result to persistent storage, one could say it did not write plaintext to persistent storage. However, the XOR does not actually help security because the XOR value resides as a constant value inside the drive.*

In some cases, the user of the product does not want to use any authorization factors, but the user still wants the product to encrypt the data he or she sends to it. The DSC must recover the SK from persistent storage when it powers up without requiring an authorization factor from the user. In the use case in which the SK is used to encrypt user data, the encryption of the data allows the user to erase the contents of the storage device very quickly by cryptographically erasing the SK. In this use case, the product recovers the SK from persistent storage without requiring an authorization factor from the user. This is equivalent to storing the SK to persistent storage without protecting, which is equivalent to storing the data (and SK?) in plaintext.

### FCS\_STG\_EXT.2 Key Storage Encryption

**FCS\_STG\_EXT.2 Key Storage Encryption**

**FCS\_STG\_EXT.2.1** The TSF shall encrypt AKs, SKs, KEKs, and [long-term trusted channel key material, all software-based key storage, no other keys] using one of the following methods: [selection: a NIST SP 800-56B key establishment scheme, symmetric encryption in [selection: AES\_CCM, AES\_GCM, AES\_KW, AES\_KWP, CAM\_CBC, CAM\_CCM, CAM\_GCM] mode as specified in FCS\_COP.1/SKC].

### FCS\_STG\_EXT.3 Key Integrity Protection

**FCS\_STG\_EXT.3 Key Integrity Protection**

**FCS\_STG\_EXT.3.1** The TSF shall protect the integrity of any encrypted AKs, SKs, KEKs, and [selection: long-term trusted channel key material, all software-based key storage, no other keys] by using [selection:

* Symmetric encryption in [selection: AES\_CCM, AES\_GCM, AES\_KW, AES\_KWP, CAM\_CCM, CAM\_GCM] mode in accordance with FCS\_COP.1/SKC;
* A hash of the stored key in accordance with FCS\_COP.1/Hash;
* A keyed hash of the stored key in accordance with FCS\_COP.1/HMAC;
* A digital signature of the stored key in accordance with FCS\_COP.1/SigVer using an asymmetric key that is protected in accordance with FCS\_STG\_EXT.2;
* An immediate application of the key for decrypting the protected data followed by a successful verification of the decrypted data with previously known information

].

**FCS\_STG\_EXT.3.2** The TSF shall verify the integrity of the [selection: hash, digital signature, MAC] of the stored key prior to use of the key.

This requirement is not applicable to derived keys that are not stored. It is not expected that a single key will be protected from corruption by multiple of these methods; however, a product may use one integrity-protection method for one type of key and a different method for other types of keys.

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

## User Data Protection

### FDP\_ACC.1 Subset Access Control

**FDP\_ACC.1 Subset Access Control**

**FDP\_ACC.1.1** The TSF shall enforce the[*Access Control SFP*] on [

* *Subjects: S.DSC, S.Admin, S.CA*
* *Objects: OB.P\_SDO and all attributes, OB.T\_SDO and all attributes, OB.AuthData, OB.PState, OB.FAACntr, OB.AntiReplay, OB.Context, [selection: [assignment: list of other objects and/or attributes], no other objects or attributes]*
* *Operations: OP.Import, OP.Create, OP.Use, OP.Modify, OP.Attest, OP.Store, OP.Export, OP.Destroy*].

The set of operations specified in the assignment can be collectively referred to as “access”. Any subsequent use of the term “access” should be interpreted to refer to one or more of these events.

### FDP\_ACF.1 Security Attribute Based Access Control

**FDP\_ACF.1 Security Attribute Based Access Control**

**FDP\_ACF.1.1** The TSF shall enforce the [*Access Control SFP*] to objects based on the following: [

1. *whether the subject is currently authorized for access (in the relevant access mode) to the SDO*
2. *whether the subject is authorized for access to the SDO according to the SDO’s re-authorization conditions*
3. *whether the subject is currently authorized to change the attributes of the SDO*
4. *[selection: [assignment: other conditions that control access to objects], no other conditions]*

].

**FDP\_ACF.1.2** The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [

1. *Only a subject who fulfills the following conditions shall be allowed to access an SDO:*
   1. *The subject is currently authorized for access of the SDO (in the relevant access mode); and*
   2. *The authorization is in accordance with the re-authorization conditions of the SDO (SDO.Reauth):*
      1. *if the re-authorization conditions have the value ‘none’ then the subject has made an initial authorization of the use of the SDO since their most recent authentication.*
      2. *if the re-authorization conditions have the value ‘each use’ then the subject has specifically authorized the current use of the SDO.*
2. *[selection: [assignment: other rules that determine the operations that are permitted based on subject-object combinations], no other rules]*]*.*

**FDP\_ACF.1.3** The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: [*assignment: rules, based on security attributes, that explicitly authorize access of subjects to objects*].

**FDP\_ACF.1.4** The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [*assignment: rules, based on security attributes, that explicitly deny access of subjects to objects*].

FDP\_ACF.1.1 Bullets 1 and 3 distinguishes between authorization to access the data or content of the SDO and the attributes associated with the content of the SDO.

FDP\_ACF.1.2 requires that an SDO can only be used when the subject requesting the access fulfills two conditions: the subject has been authorized for the access mode required as in this SFR and the access is authorized in accordance with the re-authorization conditions attribute of the SDO (see FIA\_UAU.6/SDO and FMT\_MSA.3/SDO).

FDP\_ACF.1.2 refers to controls over changing attributes that are specified in more detail in FMT\_MSA.1/SDO.

The DSC may contain pre-installed SDOs. The DSC will enforce access control for pre-installed SDOs like any other SDO it contains or manages.

### FDP\_APW\_EXT.1 Storage of Authentication Tokens

**FDP\_APW\_EXT.1 Storage of Authentication Tokens**

**FDP\_APW\_EXT.1.1** The TSF shall protect the confidentiality of [selection: authorization values, authentication tokens, data derived from authenticators] prior to storing them.

FDP\_APW\_EXT.1 applies regardless of whether the TSF stores the authorization values and authentication tokens as SDEs or attributes in an SDO, and whether the TSF stores the SDO internal or external to the TOE.

Regardless of how authorization values and authentication tokens are used, the TSF may store them in an encrypted format, or as data derived from an authentication token such as a hash, as long as it does not store the data plaintext as per FPT\_APW\_EXT.1.

If the TSF stores this data internal to the TOE, it must use FMT\_MSA.3/SDO to protect its confidentiality.

If the TSF stores this data external to the TOE, it must use FDP\_ETC\_EXT.1 to protect its confidentiality.

### FDP\_ETC\_EXT.1 Secure Propagation of User Data

**FDP\_ETC\_EXT.1 Secure Propagation of User Data**

**FDP\_ETC\_EXT.1.1** The TSF shall propagate [selection: wrapped authorization values, wrapped SDOs] such that [selection: only the TSF can unwrap the data, only authorized users can unwrap the data].

FDP\_IFF.1.2 applies when propagating SDOs for storage on the platform local to the DSC (i.e. only the TSF can unwrap the data) and when propagating SDOs for consumption by another device remote to the DSC and potentially remote to the platform containing the DSC (i.e. only authorized DSCs can unwrap the data).

### FDP\_FRS\_EXT.1 Factory Reset

**FDP\_FRS\_EXT.1 Factory Reset**

**FDP\_FRS\_EXT.1.1** The TSF shall permit a factory reset of the TOE upon: [selection: activation by external interface, presentation of [*assignment: type(s) of authorization data required and reference to their specification*], no actions or conditions].



If the DSC provides factory reset and requires an authorization to carry out the operation then the ST author selects either ‘presentation of…’ and fills in the authorization data accepted (e.g. a PIN or a cryptographic token based on some specification referenced in the assigned value). If the DSC provides factory reset external to the DSC without requiring authorization then the ST author selects ‘activation by external interface’ this selection is intended for use when the device containing the DSC takes responsibility for obtaining and checking the authorization for factory reset.

### FDP\_IFC.1 Subset Information Flow Control

**FDP\_IFC.1 Subset Information Flow Control**

**FDP\_IFC.1.1** The TSF shall enforce the [*Flow Control SFP*] on [

* *Subjects: S.CA, S.DSC, S.EPS*
* *Information: OB.T\_SDO and all attributes, OB.Context, OB.AuthData, OB.PState, OB.FAACntr, OB.AntiReplay, [selection: EPS.ID, [assignment: list of other information], no other information]*
* *Operations: OP.Import, OP.Attest, OP.Export*].

### FDP\_IFF.1 Simple Security Attributes

**FDP\_IFF.1 Simple Security Attributes**

**FDP\_IFF.1.1** The TSF shall enforce the [*Flow Control SFP*] based on the following types of subject and information security attributes: [

1. *Whether channels selected to perform the OP.Import operations against SDEs are secure*
2. *[selection: [assignment: other types of subject and information security attributes], no other attributes]*].

**FDP\_IFF.1.2** The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold: [

1. *The TSF creates a new SDO (OP.Create) using SDEs parsed only over a secure channel.*
2. *The TSF creates a new SDO (OP.Create) by binding one or more parsed and generated SDEs with a set of attributes for the SDO*
3. *The TSF shall only parse an SDO (OP.Import) over a secure channel and in a protected form*
4. *The TSF shall propagate an SDO (OP.Export) if all of the following are true:*
   1. *The SDO has an Export Flag value marking it as able to be propagated*
   2. *A user authorizes propagation of the SDO according to the re-authorization conditions for the SDO*
   3. *The TSF encrypts the SDO under a wrapping key*
5. *The TSF shall propagate SDOs (OP.Export) over a secure channel*
6. *[selection: [assignment: other rules], no other rules]*]*.*

A secure channel for parsing of SDEs and/or SDOs is one that meets the requirements of FPT\_ITC\_EXT.1 or FPT\_ITP\_EXT.1.

A newly created SDO consists of one or more SDEs where each SDE has been either parsed over a secure channel, or else generated on the DSC itself. The SDEs are bound together with a set of attributes initialized as specified in FMT\_MSA.3/SDO. The cryptographic binding operation used to create the SDO is as defined in FCS\_COP.1/Bind, and is used to create the Binding Data attribute of the SDO.

Where an SDO is parsed into the DSC as in FDP\_IFF.1 then this uses a protected form described by FCS\_COP.1/KeyEnc (e.g. KW, KWP). The DSC can therefore only parse SDOs for which it holds the relevant KEK.

The Security Target and/or Operational Guidance is expected to specify how the DSC distinguishes SDEs and SDOs when Parsing.

**FDP\_IFF.1.3** The TSF shall enforce [*no additional information flow control rules*].

**FDP\_IFF.1.4** The TSF shall explicitly authorize an information flow based on the following rules: [*assignment: rules, based on security attributes that explicitly authorize information flows*].

**FDP\_IFF.1.5** The TSF shall explicitly deny an information flow based on the following rules: [*a parsed SDO shall be rejected by the TSF if it cannot confirm the SDO’s integrity*]*.*

When an SDO is parsed then its integrity is checked on import as in FDP\_SDI.2.

### FDP\_ITC.1 Import of User Data without Security Attributes

**FDP\_ITC.1 Import of User Data without Security Attributes**

**FDP\_ITC.1.1** The TSF shall enforce the [*Flow Control SFP*] when importing user data, controlled under the SFP, from outside of the TOE.

**FDP\_ITC.1.2** The TSF shall ignore any security attributes associated with the user data when imported from outside the TOE.

**FDP\_ITC.1.3** The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: [*assignment: additional importation control rules*].

This SFR normally applies to user data (i.e. SDEs, maybe SDOs) that arrive with no security attributes. Depending on the functionality of the TSF, the SFP could enforce access control rules by having the TSF apply security attributes as it stores the user data. This does not contradict FDP\_ITC.2.

### FDP\_ITC\_EXT.1 Parsing of SDEs

**FDP\_ITC\_EXT.1 Parsing of SDEs**

**FDP\_ITC\_EXT.1.1** The TSF shall support importing SDEs using [selection: physically protected channels as specified in FTP\_ITP\_EXT.1, encrypted data buffers as specified in FTP\_ITE\_EXT.1, cryptographically protected data channels as specified in FTP\_ITC\_EXT.1].

**FDP\_ITC\_EXT.1.2** The TSF shall verify the integrity of the SDE using [*assignment: list of ways the TSF verifies the integrity of SDEs*].

**FDP\_ITC\_EXT.1.3** The TSF shall bind SDEs to security attributes using [*assignment: list of ways the TSF generates security attributes and binds them to the SDEs*].

The way the TSF checks the integrity of the SDE depends on the method of importation. For example, the encrypted data channel may provide data integrity as part of its service.

When a TSF parses an SDE, it should generate security attributes and create an SDO by binding the security attributes to the SDE.

### FDP\_ITC.2 Import of User Data with Security Attributes

**FDP\_ITC.2 Import of User Data with Security Attributes**

**FDP\_ITC.2.1** The TSF shall enforce the [*Flow Control SFP*] when importing user data, controlled under the SFP, from outside of the TOE.

**FDP\_ITC.2.2** The TSF shall use the security attributes associated with the imported user data.

**FDP\_ITC.2.3** The TSF shall ensure that the protocol used provides for the unambiguous association between the security attributes and the user data received.

**FDP\_ITC.2.4** The TSF shall ensure that interpretation of the security attributes of the imported user data is as intended by the source of the user data.

**FDP\_ITC.2.5** The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: [*assignment: additional importation control rules*].

This SFR normally applies to user data (i.e. SDEs, maybe SDOs) that arrive with security attributes. Depending on the functionality of the TSF, the SFP could enforce access control rules by having the TSF apply additional security attributes as it stores the user data. This does not contradict FDP\_ITC.1.

### FDP\_ITC\_EXT.2 Parsing of SDOs

**FDP\_ITC\_EXT.2 Parsing of SDOs**

**FDP\_ITC\_EXT.2.1** The TSF shall support importing SDOs using [selection: physically protected channels as specified in FTP\_ITP\_EXT.1, encrypted data buffers as specified in FTP\_ITE\_EXT.1, cryptographically protected data channels as specified in FTP\_ITC\_EXT.1].

**FDP\_ITC\_EXT.2.2** The TSF shall verify the integrity of the SDO using [*assignment: list of ways the TSF verifies the integrity of SDOs*].

The way the TSF checks the integrity of the SDO depends on the method of importation. For example, the encrypted data channel may provide data integrity as part of its service.

When a TSF parses an SDO, it should already have a set of security attributes. However, the TSF may modify these attributes, if authorized, to comply with security policies on the TOE.

### FDP\_MFW\_EXT.1 Mutable/Immutable Firmware

**FDP\_MFW\_EXT.1 Mutable/Immutable Firmware**

**FDP\_MFW\_EXT.1.1** The DSC contains [selection: immutable, mutable] firmware.

*The ST author should include FDP\_MFW\_EXT.2, FPT\_FLS.1/FW, and FPT\_RPL.1/Rollback if-and-only-if “mutable” is selected.*

### FDP\_RIP.1 Subset Residual Information Protection

**FDP\_RIP.1 Subset Residual Information Protection**

**FDP\_RIP.1.1** The TSF shall ensure that any previous information content of a resource is made unavailable upon the [*deallocation of the resource from*] the following objects: [

* *Security Data Objects*
* *Security Data Elements*]*.*

When an SDE is a key then it is also subject to the key destruction requirements in the various iterations of FCS\_CKM.4, depending on where and how it is stored.

### FDP\_SDC\_EXT.1 Confidentiality of SDEs

**FDP\_SDC\_EXT.1 Confidentiality of SDEs**

**FDP\_SDC\_EXT.1.1** The TSF shall use [selection: protected storage, symmetric encryption, key wrapping] to protect the confidentiality of [*assignment: list of internally and externally stored SDEs identified in the Confidential SDE List attribute of an SDO*].

**FDP\_SDC\_EXT.1.2** The TSF shall use [selection: FCS\_CKM\_EXT.5, FCS\_CKM.1/SK, FCS\_CKM.1/KEK] to derive or generate the key to encrypt the SDEs.

*This SFR applies to confidential SDEs, especially secret and private keys, Allowed Random Number Generators’ state data, and vendor verification reference data.*

If the TOE stores these parameters outside of its boundary, it must encrypt them according to the cryptographic requirements for key encryption.

Vendor pre-installed SDOs includes both objects installed during manufacturing, and those provisioned by the vendor before final release to customer. The manufacturer administrator and no one else owns and controls these objects.

The confidential-SDE List attribute of the SDO indicates those SDEs that require confidentiality. If SDEs do not require confidentiality, then its omission from this list indicates that confidentiality is not required.

### FDP\_SDI.2 Stored Data Integrity Monitoring and Action

**FDP\_SDI.2 Stored Data Integrity Monitoring and Action**

**FDP\_SDI.2.1** The TSF shall monitor SDOs and SDEs controlled by the TSF for [*integrity errors*] on all objects, based on the following attributes: **[selection: protected storage, cryptographic hash, digital signature].**

**FDP\_SDI.2.2** Upon detection of a data integrity error, the TSF shall [

* *prohibit the use of the altered data*
* *send notification of the error to the user*].

This SFR deals with the mechanism that protects the integrity of the SDEs and security attributes within an SDO. This provides the binding data to ensure that prevents unauthorized changes to the SDEs and attributes.

The Cryptographic requirements for cryptographic hashes and digital signatures apply here.

No specific requirement is placed here on the nature of the integrity protection data, but the Security Target shall describe this protection measure, and shall identify the iteration of FCS\_COP.1/HASH or FCS\_COP.1/HMAC that covers any cryptographic algorithm used.

The integrity protection data in FDP\_SDI.2.1 is included in the list of attributes identified in FMT\_MSA.1/SDO, and protects the value of the SDEs and of the SDO security attributes.

When an SDO is parsed then its integrity is checked when it is imported into the TOE.

## Identification and Authentication

When a platform process requests the ability to create, use, modify, dispose of, etc., an SDE or SDO within the DSC, as a matter of policy, the DSC may expect or request authorization from the platform process, which may include authentication of the requester on whose behalf the platform process is acting. The DSC assumes the requester to be either a person, a process, or a device. The rules on how the requester formats the request will be outside the scope of this cPP. Upon request (or as a matter of an established protocol), the interface (on behalf of the user) presents to the DSC process those authorization values required to authorize execution of the event request. This may include one or more different types of authentication credentials. The DSC validates these items before acting upon the requested event. The validation may simply compare the authorization values to an expected value, or perform a more complex cryptographic protocol to verify the authenticity of the user. After validation, the DSC may then create and subsequently use an authorization value to represent the validation of these authorization values in anticipation of future requests.

Requirements covering the generation of specific authorization values, such as X.509 certificates and biometric templates, are outside the scope of the DSC. In addition, requirements related to the strength, quality, and performance of authorization values are all outside the scope of the DSC and are expected to be met by the platform, where applicable.

### FIA\_AFL\_EXT.1 Authorization Failure Handling

**FIA\_AFL\_EXT.1 Authorization Failure Handling**

**FIA\_AFL\_EXT.1.1** The TSF shall maintain [selection: a unique counter for [selection, choose one of: each SDO, the following SDOs [*assignment: list of SDOs*]], one global counter covering [selection, choose one of: all SDOs, the following SDOs [*assignment: list of SDOs*]]], called the failed authorization attempt counter(s), that counts of the number of unsuccessful authorization attempts that occur related to authorizing access to these SDOs.

**FIA\_AFL\_EXT.1.2** The TSF shall maintain a [selection, choose one of: static, administrator configurable variable] threshold of the minimal acceptable number of unsuccessful authorization attempts that occur related to authorizing access to these SDOs.

**FIA\_AFL\_EXT.1.3** When the failed authorization attempt counter(s) [selection, choose one of: meets, surpasses] the threshold for unsuccessful authorization attempts, the TSF shall [selection, choose one of:

* prevent future authorization attempts for a static prescribed amount of time;
* prevent future authorization attempts for an administrator configurable amount of time;
* prevent all future authorization attempts indefinitely (i.e., lock), as described by FIA\_AFL\_EXT.2;
* factory reset the TOE wiping out all user SDOs, as described by FDP\_FRS\_EXT.2

] for these SDOs.

The product validates the authorization factor(s) prior to allowing the user access to the SDE/SDO. In cases with validation of the authorization factor(s) fails, the product will not allow access to SDE/SDO. The product validates the authorization factor(s) in such a way that it does not allow an attacker to circumvent the other requirements to gain knowledge about the SDE/SDO or other keying material that protects them from inadvertent exposure.

### FIA\_SOS.2 TSF Generation of Secrets

FIA\_SOS.2 TSF Generation of Secrets

**FIA\_SOS.2.1** The TSF shall provide a mechanism to generate **authorization values** that meet [*the following quality metrics:*

* *For each authentication attempt, the probability shall be less than one in 1,000,000 that a random attempt will be successful*
* *For multiple attempts to authenticate during a one-minute period, the probability shall be less than one in 100,000 that a series of random attempts will be successful*].

**FIA\_SOS.2.2** The TSF shall be able to enforce the use of TSF generated **authorization values** for [*assignment: non-empty list of TSF functions*].

*This SFR expects the TSF must generate authorization values from a sufficiently large key space to ensure that users cannot employ random guessing as a statistically plausible method of authorizing actions within the TOE, both for a single event and over a session.*

*As an example, consider this theoretical authorization in which a PIN is used to authenticate a user: a TOE that uses a six-digit PIN and enforces a ten-second delay for each failed PIN entry made after the fifth consecutive attempt (assuming that a PIN authentication attempt takes two seconds). The odds of correctly guessing the PIN on a single attempt are 1,000,000:1 and the odds of correctly guessing the PIN if given access to it for a period of one minute is approximately 100,000:1. This method would therefore meet the minimum quality metrics specified in this SFR along with FIA\_AFL\_EXT.1.*

### FIA\_UIA\_EXT.1 User Identification and Authentication before Any Action

**FIA\_UIA\_EXT.1 User Identification and Authentication before Any Action**

**FIA\_UIA\_EXT.1.1** The TSF shall successfully identify and authenticate each user using [selection: PIN/try-PIN comparison, salted hash comparison, unsalted hash comparison, [*assignment: other comparison mechanism*]] before authorizing any other TSF-mediated actions on behalf of that user.

This requirement specifies the TSF exercise an authentication mechanism by which the product authenticates the identity of the user and authorizes it to operate with SDOs. A user could present a unique authorization factor. In either case, the product validates the authorization factor(s) prior to granting the user authorization to perform the requested operation with the Security Data Object. Furthermore, the authentication or authorization validation methods do not compromise the contents of the SDOs. The means of validation may vary based on the type of authentication factor(s).

The product may accept authorization factors with no further conditioning.

The product may authorize the user only once; it does not have to authorize the user each time he or she tries to operate with the SDO.

An authenticator is shortened name for authentication token. An authentication token is critical data bound to a user. Such data, when presented to the TOE and successfully verified by it, authenticates the user. The TOE may use the successful authentication of a user as an authorization to execute an action on its behalf, or to perform a requested operation on or with an SDO.

### FIA\_UAU.6/SDO Re-Authenticating (Access to SDO)

**FIA\_UAU.6/SDO Re-Authenticating (Access to SDO)**

**FIA\_UAU.6.1/SDO**The TSF shall **authorize and re-authorize**[[3]](#footnote-4) the user **for access to an SDO** under the conditions: [

1. *Initial authorization (for the first access to the SDO in the current session) by successful completion of one of the following authorization methods: [assignment: list of authorization methods]; and*
2. *Re-authorization, by further successful completion of one of the authorization methods in condition 1 above, in accordance with value of the ‘re-authorization conditions’ attribute of the SDO as follows:*
   1. *if the re-authorization conditions have the value ‘none’ then the subject has made an initial authorization of the use of the SDO since the start of the current session;*
   2. *if the re-authorization conditions have the value ‘each access’ then the subject has specifically authorized the current use of the SDO;*
   3. *if the re-authorization conditions specify a list of other events then either none of the events has occurred since successful initial authorization of the use of the SDO by the subject, or else the subject has specifically authorized the use of the SDO since the last event.*

].

*The DSC supports the authorization of a user to perform an operation (generate, use, modify, propagate, destroy) on an SDO.*

*The allowed values for the re-authorization conditions attribute of an SDO are defined in FMT\_MSA.3/SDO and the SDO Attributes Initialization Table. The rules in FDP\_ACF.1.2 and FDP\_IFF.1.2 also ensure that the need for re-authorization has been checked before access to an SDO.*

### FIA\_USB\_EXT.1 Response to User Subject Binding

**FIA\_USB\_EXT.1 Response to User Subject Binding**

**FIA\_USB\_EXT.1.1** Upon binding a user to access an SDO, [selection, choose one of: the security attributes of the SDO shall remain unchanged, the TSF shall change the values of the security attributes of the SDO only if the user is authorized to change them].

When the TSF binds a user to access an SDO, this means that the TSF has authenticated the user and that the TSF authorized the user to have the right to exercise one or more of the following actions: generate the SDO, modify the SDO, including its security attributes, use the SDO in a TOE operation, propagate or duplicate the SDO for use by a device external to the DSC, or destroy the SDO. The user may not have exclusive rights to exercise the operations listed.

Users in this context means both administrative and regular users. Policy as represented by the attributes in the SDO dictates whether or not a user must authenticate itself in order to authorize access to the SDO.

It is possible that the attributes of some SDOs should remain unchanged, and that the attributes of other SDOs may be changed by authorized users. If this is the case, then the ST author should iterate this SFR and indicate in the TSS which SDOs apply to each iteration.

## Security Management

### FMT\_MOF\_EXT.1 Management of Security Functions Behavior

**FMT\_MOF\_EXT.1 Management of Security Functions Behavior**

**FMT\_MOF\_EXT.1.1** The TSF shall restrict the ability to perform the functions in FMT\_SMF.1 to authenticated administrators.

### FMT\_MSA.1/SDO Management of Security Attributes (Secure Data Objects)

**FMT\_MSA.1/SDO Management of Security Attributes (Secure Data Objects)**

**FMT\_MSA.1.1/SDO** The TSF shall enforce the [*Access Control SFP*] to restrict the ability to [modify] the security attributes **[*assignment*: *list of security attributes, to include attributes as specified in Table 12 below*]** to **[*assignment*: *list of subjects, objects, and operations among subjects and SDOs to include at least the constraints specified in Table 12 below*].**

The SDO Attributes Modification Table defines the required constraints on security attribute modification. The Security Target completes the other parts not specified here (along with any other information for other security attributes relevant to a particular TOE).

The assignments of authorized subjects in the SDO Attributes Modification Table may be defined by the ST author in terms of roles or in terms of an action such as presentation of a valid authorization token of a particular type (in this case the ST author identifies in an Application Note the other SFRs that govern the action).

The TSF vendor may pre-install SDOs with default attributes. The Security Target should make clear which attributes the administrators may change or are prohibited from changing. It should also make clear between authorization values required to use pre-installed SDOs and authorization values required to change the attributes of pre-installed SDOs.

The SDO Attributes Modification Table lists SDO ID as “cannot be modified”. In some cases, a change in the attributes may cause a change in the SDO ID. In these cases, a change in the SDO ID causes the creation of a new SDO and possibly the loss of the old SDO.

Only authorized subjects can change the attributes of an SDO, and only as permitted in the SDO Attributes Modification Table.

| SDO Attribute | Modification Constraints |
| --- | --- |
| SDO.ID | Cannot be modified |
| SDO.Type | Cannot be modified |
| SDO.AuthData | [*assignment*: *list of subjects that are authorized to modify SDO reference authorization data*] |
| SDO.Reauth | [*assignment*: *list of subjects that are authorized to modify re-authorization conditions*] |
| SDO.Conf | [*assignment*: *list of subjects that are authorized to modify confidential SDE-list*] |
| SDO.Export | [*assignment*: *list of subjects that are authorized to modify export flag*] |
| SDO.Integrity | Cannot be modified by users (maintained automatically by TSF) |
| SDO.Bind | Cannot be modified by users (maintained automatically by TSF) |

Table 12: Supported Methods for SDO Attributes Security Attributes

### FMT\_MSA.3/SDO Static Attribute Initialization (Secure Data Objects)

This SFR deals with the initialization of the attributes of an SDO when it is created by parsing or provisioning. The generation process includes SDOs created by the TSF (provisioned) and those imported via FDP\_ITC.2 (parsed).

The TSF is expected to give an SDO a set of security attributes at the time of its creation. This set is expected to include at least the following attributes:

* SDO identifier
* SDO type
* SDO reference authorization data (i.e. the data that is used when determining whether to grant access to an SDO, for each relevant mode of access, on the basis of an authorization token presented to the DSC)
* Re-authorization conditions (i.e. event after which re-authorization is required)
* Confidential-SDE list (each SDE in this list is held encrypted when the SDO is stored)
* Export Flag (indicating whether the SDO is allowed to be propagated)
* Integrity protection data
* Binding Data (created by the TOE to strongly link or associate the SDO with other entities such as the TOE itself or with other SDOs in a hierarchy such as a child to a parent).

The TSF provides the capability to protect the contents of an SDO, i.e. the set of its SDEs together with the SDO attributes, from unauthorized modification. The DSC shall check for such modifications before using the SDO or any of its SDEs.

**FMT\_MSA.3/SDO Static Attribute Initialization (Secure Data Objects)**

**FMT\_MSA.3.1/SDO** The TSF shall enforce the [*Access Control SFP*] to provide [selection, choose one of: restrictive, permissive*,* [*assignment: other property*]] default values for security attributes that are used to enforce the SFP.

**FMT\_MSA.3.2/SDO** The TSF shall allow the [*authorized identified roles, according to the constraints in* Table 13 *below*] to specify alternative initial values to override the default values when an object or information is created.

| SDO Attribute | Initialization Constraints |
| --- | --- |
| SDO.ID | Initialized by generation process |
| SDO.Type | Initialized by generation process |
| SDO.AuthData | Initialized by creator during generation |
| SDO.Reauth | Initialized by creator during generation to one of the values [selection: none, each use, [*assignment: events triggering re-authorization*]] |
| SDO.Conf | Initialized by creator during generation |
| SDO.Export | Initialized by generation process |
| SDO.Integrity | Initialized automatically by TSF |
| SDO.Bind | Initialized by generation process |

Table 13: Supported Methods for Secure Data Objects Attributes Initialization

The SDO Attributes Initialization Table is referenced from FMT\_MSA.3/SDO and matches the attributes covered by FMT\_MSA.1/SDO (which defines controls on the modification of the attributes). The initialization of these security attributes occurs when an SDO is either parsed by the TOE or generated on the TOE. The required constraints on security attribute initialization specified in this PP are shown in “Table 5: SDO Attributes Initialization Table”; the Security Target completes the selection and assignments in the SFR and adds to the table any other information for other security attributes relevant to a particular TOE.

The ’SDO reference authorization data’ is data that is required in order to validate authorization of a subject to access the SDO (in each of the modes relevant to that SDO). The nature of this data will depend on the authorization mechanism used in the TOE, as described in FIA\_UIA\_EXT.1.

The ‘re-authorization conditions’ attribute for an individual SDO takes one of the values defined in the selection in the re-authorization conditions row of the SDO Attributes Initialization Table (this selection defines the re-authorization conditions that the TOE supports). Examples of TOE-specified events might be explicit revocation of authorization by a user or process, expiry of a time interval, or completion of a fixed number of uses since the last authorization. The re-authorization conditions are used in FIA\_UAU.6/SDO, FDP\_ACF.1, and FDP\_IFF.1. These determine whether a single authorization by the SDO owner will allow any number of uses of the SDO until the end of the user’s session (value ‘none’), or whether each use of the SDO must be individually authorized (value ‘each access’), or whether re-authorization must happen each time one of the TOE-specified events occurs.

The Confidential-SDE list indicates which SDEs, if any, the TOE should encrypt when not in operational use. The TOE should use the methods in FCS\_COP.1/SKC, FCS\_STG\_EXT.1, or FCS\_STG\_EXT.2 to protect the SDEs in this list.

The ‘Export Flag’ attribute takes one of the values ‘exportable’ or ‘non-exportable’.

The ‘Integrity Protection Data’ attribute includes evidence that the TSF can use to protect and verify the integrity of the SDO.

Attributes assigned by the TOE to any parsed SDOs must be described in the Security Target and in operational user guidance.

The TOE uses the Binding Data for an SDO to strongly link the SDO to the TOE, a parent SDO in a hierarchy, or to nothing at all. SDOs bound to nothing may freely travel from one TOE to another without restrictions. If bound to another SDO as a child to a parent in a hierarchy, it may travel only where the parent SDO travels. If bound to the TOE, it may travel to any other TOE for any reason, even if the TOE moves its parent to another TOE. Note that vendors will initialize attributes of pre-installed SDOs with default values. However, authorization values to change the attributes of pre-installed SDOs may differ from the authorization value required to use the pre-installed SDO.

The vendor should document the implicit attributes for pre-installed SDOs and SDOs stored in special locations.

In cases in which the SDO ID is a cryptographic hash of the attributes and SDEs, that value represents both the ID and projects the integrity of the SDO, including the SDEs. As the TOE unwraps an incoming SDO, it may automatically check the integrity. For pre-installed SDOs in protected storage, the hardware plus the TSF projects the integrity of them.

When a remote peer sends an SDO to the TOE, it properly indicates through the SDE-confidentiality list of any authorization values and authentication tokens present in the SDO, whether they are present in the SDE or as attributes, which control access to the SDE.

When a TOE generates an SDO internally for the first time, it properly indicates through the SDE-confidentiality list any SDEs that are authorization values or authentication tokens. Similarly, if any of the attributes are authorization values or authentication tokens, the TOE will properly indicate through the SDE-confidentiality list that it will encrypt them prior to storing them.

The TOE may contain pre-installed SDOs or SDOs either provisioned the first time the user turns on the TOE or provisioned as the result of a “factory reset” event. TSFs may refer to such persistent SDOs as root keys or trusted anchors. Pre-installed SDOs may reside in immutable hardware and persist across factory resets. Other persistent SDOs may persist until a user issues a “factory reset” which either cryptographically erases the SDOs or overwrites them by provisioning new ones. These SDOs may not contain a confidential SDE list since either these persistent values serve as a root encryption key for a hierarchy of SDOs, or they serve as a KDF seed for generating root encryption keys for a hierarchy of SDOs.

It is possible that the default attributes of some SDOs should be restrictive, and that the default attributes of other SDOs may be permissive. If this is the case, then the ST author should iterate this SFR and indicate in the TSS what the default attribute properties are for each SDO.

### FMT\_SMF.1 Specification of Management Functions

**FMT\_SMF.1 Specification of Management Functions**

**FMT\_SMF.1.1** The TSF shall be capable of performing the following management functions: [

* *Set authorization failure parameters for FIA\_AFL\_EXT.1*
* *Update TOE firmware and pre-installed SDOs, if mutable*
* *Reset TOE to factory state for FDP\_FRS\_EXT.1*
* *Configure authorization policies*].

Recall that resetting a TOE to factory state also wipes all user data. Configuring authorization policies includes setting policies for allowed access to SDOs.

Protections for pre-installed SDEs/SDOs come through the firmware, and by extension, through firmware updates. In the same vein, the authorized updates may also affect the SDEs as well, if the vendor so chooses. One could say that the authorized update binds the attributes present in the functionality of the firmware to the pre-installed SDEs.

### FMT\_SMR.2 Separation of Roles

**FMT\_SMR.2 Restrictions on Security Roles**

**FMT\_SMR.2.1** The TSF shall maintain the roles: [*administrator, user*].

**FMT\_SMR.2.2** The TSF shall be able to associate users with roles**.**

**FMT\_SMR.2.3** The TSF shall ensure that the conditions [

* *Only users can access their own encrypted data,*
* *Only administrators can perform privileged functions*]

are satisfied.

## Protection of the TSF

### FPT\_APW\_EXT.1 Protection of Administrator Passwords

FPT\_APW\_EXT.1 Protection of Administrator Passwords

**FPT\_APW\_EXT.1.1** The TSF shall store TOE authenticators in non-plaintext form.

**FPT\_APW\_EXT.1.2** The TSF shall prevent the reading of plaintext TOE authenticators.

*The intent of the requirement is that the TOE does not store its raw authentication tokens (e.g. PIN, biometric data) in the clear. The TOE prevents non-Administrators from reading plaintext TOE authenticators.*

### FPT\_FLS.1/FI Failure with Preservation of Secure State (Fault Injection)

FPT\_FLS.1/FI Failure with Preservation of Secure State (Fault Injection)

**FPT\_FLS.1.1/FI** The TSF shall preserve a secure state when the following types of failures occur: [*fault injections*].

*Note that a secure state does not imply the uninterrupted enforcement of all claimed security functionality it is appropriate for the TSF to “fail closed” and block the execution of security-relevant behavior if a fault injection attempt or other significant glitch occurs.*

### FPT\_MOD\_EXT.1 Debug Modes

FPT\_MOD\_EXT.1 Debug Modes

**FPT\_MOD\_EXT.1.1** The TSF shall provide no access to debug modes.

*‘Debug modes’ may include, but are not limited to, any alternate mode of operation, such as developer mode, test mode, manufacturer mode, or altered boot mode.*

### FPT\_PHP.3 Resistance to Physical Attack

FPT\_PHP.3 Resistance to Physical Attack

**FPT\_PHP.3.1** The TSF shall use [selection: METHOD 1, METHOD2] to detect physical tampering and respond by [selection: ACTION 1, ACTION 2] when detecting physical tampering.

*Physical protection mechanisms as envisioned by this requirement are mechanisms that protect communications to the extent that encryption or other logical protections are not required to ensure confidentiality, integrity, and assured identification of endpoints. Such mechanisms may include, for example, physically isolated traces, or mechanisms that take advantage of physical properties of signals to ensure that communications are receivable only by the intended endpoint.*

*Any physical external casing or potting material of the TOE is considered an ‘external interface’, not just those interfaces over which data is transmitted. This ensures that the TSF will respond appropriately if, for example, an attacker penetrates the physical surface of the DSC in an attempt to access its stored data.*

### FPT\_PRO\_EXT.1 Root of Trust

FPT\_PRO\_EXT.1 Root of Trust

**FPT\_PRO\_EXT.1.1** The DSC shall contain a Security Data Object that contains the identity of the Root of Trust.

*Every DSC is expected to have a single Root of Trust that comprises the DSC hardware and pre-installed Security Data Objects, from which services (e.g. Storage, Authorization, etc.) can be offered.*

*Depending on the use case and the way status registers are utilized, unique identity keys may be bound to either the TOE, TOE platform or both.*

*The sole presence of unique identity keys linking to the Root of Trust does not prove authenticity without the use of digital signatures.*

**FPT\_PRO\_EXT.1.2** The code and/or data of the Root of Trust shall be [selection: immutable, mutable if and only if its mutability is controlled by a unique identifiable owner].

One expects that only authorized sources can modify the single Root of Trust, such as through a secure update. A pre-installed SDO may contain the identity of the manufacturer of the RoT.

The process of authenticating the source of a secure update may involve querying the identity of the manufacturer, contained on a pre-installed SDO. If this identity is in the form of an X.509 certificate containing a signature verification key signed by the manufacturer, then the authentication process is sufficient.

A unique identifiable owner is assumed to be one with an administrative role, such as the manufacturer administrator or owner administrator; however, there may be circumstances where the owner does not take on an administrative role, which should be documented.

### FPT\_ROT\_EXT.1 Root of Trust Services

FPT\_ROT\_EXT.1 Root of Trust Services

**FPT\_ROT\_EXT.1.1** The TSF shall provide a Root of Trust for Storage, a Root of Trust for Authorization, and [selection: Root of Trust for Measurement, Root of Trust for Reporting, no others].

This document uses the GlobalPlatform definitions for Root of Trust for Storage (denoted as the Root of Trust for Authentication), Authorization, Measurement, and Reporting. DSCs use Roots of Trust for Storage to protect Security Data Objects (SDOs). Section 6.5 has a number of requirements for ensuring the TSF has functionality to authorize a user in order to access an SDO, including FIA\_UAU.6/SDO. In this context, the term “user” may refer to, but is not limited to, a human user or process. FDP\_DAU.1/Prove should be selected if-and-only-if Root of Trust for Measurement and Root of Trust for Reporting are both selected in FPT\_ROT\_EXT.1.

### FPT\_ROT\_EXT.2 Root of Trust for Storage

FPT\_ROT\_EXT.2 Root of Trust for Storage

**FPT\_ROT\_EXT.2.1** The Root of Trust for Storage shall prevent unauthorized access to SDOs.

*TOEs may use shielded locations or cryptographic protections to prevent unauthorized access to SDOs. Use FDP\_SDI.2 to protect the integrity of SDOs stored in the Root of Trust for Storage.*

### FPT\_RPL\_EXT.1 Replay Prevention

**FPT\_RPL\_EXT.1 Replay Prevention**

**FPT\_RPL\_EXT.1.1** The TSF shall have a mechanism for preventing replay of user authorization of operations on SDOs using the following methods [selection: monotonic counters, random nonces, [*assignment: other methods as specified*]].

**FPT\_RPL\_EXT.1.2** The TSF shall detect replay for the following actions: [*authorization of operations on SDOs*].

**FPT\_RPL\_EXT.1.3** The TSF shall deny the requested operation on the SDO when it detects a replay.

The TSF receives authorization from an external source to the DSC to perform an operation on an SDO. If the operation on the SDO is restricted to authorized users, then anyone observing the communication to the DSC can copy the authorization and replay it. Random nonces and monotonic counters are but two mechanisms the TSF can use to mitigate replay. In this requirement, operations on SDOs include generating, using, modifying, propagating, and destroying. Besides monotonic counters and random nonces, the TSF could employ other methods to prevent replay of user authorizations, which this document should describe.

This requirement does not specify how TSF detects replays.

### FPT\_STM.1 Reliable Time Stamps

FPT\_STM.1 Reliable Time Stamps

**FPT\_STM.1.1** The TSF shall be able to provide reliable time stamps.

*It is acceptable for the TSF to provide timestamp data either through an internal clock or a counter. It is also permissible for the TSF to obtain time data from a clock contained within the same physical enclosure as the TOE.*

### FPT\_TST.1 Integrity Checking

FPT\_TST.1 TSF Integrity Checking

**FPT\_TST.1.1** The TSF shall run a suite of self-tests **during initial start-up,** [selection: periodically during normal operation, at the request of the authorized user, **at no other condition,** at the conditions [*assignment: conditions under which self-test should occur*]] to demonstrate the correct operation of [the TSF].

**FPT\_TST.1.2** The TSF shall provide authorized users with the capability to verify the integrity of [TSF data].

**FPT\_TST.1.3** The TSF shall provide authorized users with the capability to verify the integrity of the [TSF].

*This requirement intends to cover integrity of the TSF functionality (i.e. runtime checks).*

*TSF integrity testing provides the ability to test the TSF’s correct operation. These tests are expected to be performed automatically and autonomously at start-up but may also be performed periodically during operation, at the request of the authorized user, or when other conditions are met. It also provides the ability to verify the integrity of TSF data and executable code.*

*All cryptographic functions come with known answer tests (KATs). In addition to verifying the integrity of the firmware executing the TSF, the DSC should also verify the integrity of any data associated with the TSF (such as constants for cryptographic algorithms) as well as performing the KATs.*

## Resource Utilization

### FRU\_FLT.1 Degraded Fault Tolerance

FRU\_FLT.1 Degraded Fault Tolerance

**FRU\_FLT.1.1** The TSF shall ensure the operation of [*protection of TSF data*] when the following failures occur: [*fault injection*].

*TSF data may be protected in response to a fault injection either by providing a method to ensure that the data remains protected or by logically destroying the data and/or any part of a key change that encrypts it. This behavior may differ based on the type of fault.*

# Security Assurance Requirements

The Security Objectives for the TOE in Section 4 were constructed to address threats identified in Section 3. The Security Functional Requirements (SFRs) in Section 5 are a formal instantiation of the Security Objectives. This cPP identifies the Security Assurance Requirements (SARs) to frame the extent to which the evaluator assesses the documentation applicable for the evaluation and performs independent testing.

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

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.

| Assurance Class | Assurance Components |
| --- | --- |
| Security Target (ASE) | Conformance Claims (ASE\_CCL.1) |
| Extended Components Definition (ASE\_ECD.1) |
| ST Introduction (ASE\_INT.1) |
| Security Objectives (ASE\_OBJ.2) |
| Derived Security Requirements (ASE\_REQ.2) |
| 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) | Labelling of the TOE (ALC\_CMC.1) |
| TOE CM Coverage (ALC\_CMS.1) |
| Tests (ATE) | Independent Testing - Conformance (ATE\_IND.1) |
| Vulnerability Assessment (AVA) | Vulnerability Survey (AVA\_VAN.1) |

Table 14: Security Assurance Requirements

## 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.

In addition to the using the ST to demonstrate that ASE\_TSS.1 has been satisfied, this cPP requires the creation of supplemental documentation to justify how the TOE satisfies certain SFRs. This documentation is separated from the ST because the required level of detail may include information that is proprietary to the developer of the TOE. The required supplemental documentation includes entropy documentation and key management documentation. The requirements for the entropy documentation are described in Appendix D of this cPP. The requirements for the key management documentation are described in the SD under the SFRs that require a detailed description of the TSF’s key management.

## ADV: Development

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

### Basic Functional Specification (ADV\_FSP.1)

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

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

## AGD: Guidance Documentation

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

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

* instructions to successfully install the TSF in that environment; and
* instructions to manage the security of the TSF as a product and as a component of the larger operational environment; and
* Instructions to provide a protected administrative capability.

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

### Operational User Guidance (AGD\_OPE.1)

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

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

### Preparative Procedures (AGD\_PRE.1)

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

## Class ALC: Life-cycle Support

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

### Labelling of the TOE (ALC\_CMC.1)

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

### TOE CM Coverage (ALC\_CMS.1)

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

## Class ATE: Tests

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

### Independent Testing – Conformance (ATE\_IND.1)

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

## Class AVA: Vulnerability Assessment

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

### Vulnerability Survey (AVA\_VAN.1)

As with ATE\_IND, the evaluator shall generate a report to document their findings with respect to this requirement. This report could physically be part of the overall test report mentioned in ATE\_IND, or a separate document. The evaluator performs a search of public information to determine the vulnerabilities that have been found in network infrastructure platforms and the implemented communication protocols in general, as well as those that pertain to the particular TOE. The evaluator documents the sources consulted and the vulnerabilities found in the report. For each vulnerability found, the evaluator either provides a rationale with respect to its non-applicability, or the evaluator formulates a test (using the guidelines provided in ATE\_IND) to confirm the vulnerability, if suitable. Suitability is determined by assessing the attack vector needed to take advantage of the vulnerability. If exploiting the vulnerability requires expert skills and an electron microscope, for instance, then a test would not be suitable and an appropriate justification would be formulated.

1. Optional Requirements
   1. Cryptographic Support
      1. FCS\_ENT\_EXT.1 Entropy for External IT Entities

FCS\_ENT\_EXT.1 Entropy for External IT Entities

**FCS\_ENT\_EXT.1** The TSF shall provide an interface to make entropy that meets FCS\_RBG\_EXT.1 available to external IT entities.

* + 1. FCS\_RBG\_EXT.2 External Seeding for Random Bit Generation

FCS\_RBG\_EXT.2 External Seeding for Random Bit Generation

**FCS\_RBG\_EXT.2.1** The TSF shall provide an interface to allow external seeding of the RBG with a bit-string of at least 256 bits before the RBG produces any output.

The last requirement requires the product to allow users to determine the source of entropy input into the RBG.

* 1. Protection of the TSF
     1. FPT\_PRO\_EXT.2 Data Integrity Measurements

FPT\_PRO\_EXT.2 Data Integrity Measurements

**FPT\_PRO\_EXT.2.1** The TSF shall be able to quantify the integrity of the data protected by the TOE by generating integrity measurements and assertions making them available to authorized entities.

*The generation of these integrity measurements and assertions is the creation of OB.PState.*

*Data protected by the TOE includes DSC firmware, DSC configuration data, and user data. DSC configuration data may include persistent SDEs or SDOs such as immutable or mutable root keys, authorization data, and authentication data (i.e. DSC.ID, OB.P\_SDO, OB.FAACntr, OB.AntiReplay, and OB.Context). User data may include transient SDEs and SDOs as well as authorization data and authentication data bound to these SDEs and SDOs (i.e. OB.T\_SDO). Integrity reporting is the process of attesting to integrity measurements (including those recorded in status registers in a DSC).*

**FPT\_PRO\_EXT.2.2** The TSF shall accumulate platform characteristics using a consistent [*assignment: description of process for accumulating platform characteristics*] process in which verified quantifiable measurements are accumulated to prove the integrity of its Security Data Objects.

*Although a platform may enter any state possible — including undesirable or insecure states – it can use platform characteristics, including integrity measurements and assertions, along with logging and reporting to accurately report the state derived from data attributing to those states. In this context, platform characteristics can include, but is not limited to, cryptographic hashes of code, security-critical configurations, register values (including status registers) and milestones, such as verification of firmware, or transitioning from a boot phase to an operational phase. A platform characteristic may also represent the state of some entity outside the DSC. A process independent from the DSC or the host containing the DSC may evaluate the platform characteristics and determine an appropriate action.*

* + 1. FPT\_ROT\_EXT.3 Root of Trust for Reporting Mechanisms

FPT\_ROT\_EXT.3 Root of Trust for Reporting Mechanisms

**FPT\_ROT\_EXT.3.1** The TSF shall be able to attest to a state as represented by platform characteristics with a Root of Trust for Reporting mechanism that uses for its cryptographically verifiable identity [selection: a resident key as specified in FIA\_UIA\_EXT.1, an alias key bound to the cryptographically verifiable identity in FIA\_UIA\_EXT.1] and using a signature algorithm as specified in FCS\_COP.1/SigGen.

*While it is possible for a group of components to share a single unique group identifier, it is important to ensure that individual components have their own unique identifiers relative to each other.*

*Resident keys or aliases are designed such that they are never visible outside the subset of DSC scope containing the Root of Trust services and are only to be used for encryption. Therefore, possession of such aliases or keys can only be proved indirectly by using it to decrypt a value that has been encrypted with a corresponding public key. In this way, these resident keys or aliases can provide for authentication based on decryption operations instead of producing a digital signature.*

*If “non-specialized cryptographic keys used for algorithms in FCS\_COP” is selected, it is expected that when used in the context of the Root of Trust for Reporting, these keys are not visible to full DSC scope as described above. While it is possible for a group of components to share a single unique group identifier, it is important to ensure that individual components have their own unique identifiers relative to each other.*

*The DSC will not expose the private portions of resident keys or aliases outside the subset of DSC scope containing the Root of Trust services. Therefore, possession of such aliases or keys can only be proved indirectly by using it to decrypt a value that has been encrypted with a corresponding public key. In this way, these resident keys or aliases can provide for authentication based on decryption operations instead of producing a digital signature.*

*The DSC responds to requests from an external entity to attest to the provenance and integrity of platform characteristics contained within the DSC.*

*Integrity reporting is the process of attesting to platform characteristics (including those recorded in status registers in a DSC). The philosophy behind integrity measurement, logging, and reporting is that a platform may enter any state possible — including undesirable or insecure states — but can still accurately report measurements derived from data attributing to those states. In this context, data can include, but is not limited to, code, security-critical configurations, values of registers, including status registers. An independent process may evaluate the integrity states and determine an appropriate response.*

1. Selection-Based Requirements

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

* 1. Cryptographic Support
     1. FCS\_CKM.1/SK Cryptographic Key Generation (Symmetric Encryption Key)

**FCS\_CKM.1/SK Cryptographic Key Generation (Symmetric Encryption Key)**

**FCS\_CKM.1.1/SK** The TSF shall generate **symmetric** cryptographic keys **[selection: key name from Table 15 below]** in accordance with a specified cryptographic key generation algorithm **[selection: cryptographic key generation algorithm from Table 15 below]** and specified cryptographic key sizes **[selection: key sizes from Table 15 below]** that meet the following: **[selection: list of standards from Table 15 below]**.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Identifier | Key Name | Cryptographic Key Generation Algorithm | Key Sizes | List of Standards |
| 1SK | SK | Direct Generation froma Random Bit Generator as specified in FCS\_RBG\_EXT.1 | [selection: 128 bit, 256 bit, 512 bit] | NIST SP 800-133 (Section 7.1) with ISO 18031 as an approved RBG in addition to those in NIST SP 800-133 (Section 5). |
| DSK [selection: identifier from Table n: Key Derivation Functions] | [selection: Key Type from Table n: Key Derivation Functions] | Derived from a Key Derivation Function as specified in FCS\_COP.1/KDF [selection: Key Derivation Algorithm from Table n: Key Derivation Functions] | [selection: key sizes from Table n: Key Derivation Functions] | [selection: List of Standards from Table n: Key Derivation Functions] |
| PBK | PBK | Derived from a Password Based Key Derivation Function as specified in FCS\_COP.1/PBKDF | [selection: key sizes as specified in FCS\_COP.1/ PBKDF] | [selection: standard(s) as specified in FCS\_COP.1/ PBKDF] |

Table 15: Supported Methods for Symmetric Encryption Key Generation

The intent of this requirement is to ensure that attackers cannot recover SKs with less than a full exhaust of the key space. This requirement explains SK generation regardless of how the DSC uses it or when it generates it. FDP\_ITC.1 and FDP\_ITC.2 provide options to import an SK. The encryption of user data that is not keying material, authentication values, or authorization values is outside the scope of this cPP. This cPP assumes that the DSC lacks the required resources to perform bulk encryption/decryption services at a suitable rate for end users. The host may use the SK for encrypting user data outside the boundaries of the DSC. On the other hand, the DSC may use the SK on behalf of the user to perform keyed hashes. In this case, all the requirements for generating, controlling access and use, and destroying the key while under the protection of the DSC apply.

The selection of key size 512 bits is for the case of XTS-AES using AES-256. In the case of XTS-AES for both AES-128 and AES-256, the developer is expected to ensure that the full key is generated using direct generation from the RBG as in NIST SP800-133 section.

The ST author selects Random Symmetric Key (RSK) if the ST supports creating keys directly from the output of the RBG without further conditioning, Derived Symmetric Key (DSK) should be selected if the ST supports key derivation functions which are usually seeded from RBG and then further conditioned to the appropriate key size, and Password-Based Key Derivation Function(s) (PBK) should be selected if the ST supports keys derived from passwords.

* 1. User Data Protection
     1. FDP\_DAU.1/Prove Data Authentication for Use with The Prove Service

**FDP\_DAU.1/Prove Data Authentication for Use with The Prove Service**

**FDP\_DAU.1.1/Prove**The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of [selection: [*assignment: list of objects or information types*] declared valid by the TSF, [*assignment: list of objects or information types*] declared valid by an authenticated user].

**FDP\_DAU.1.2/Prove**The TSF shall provide [*assignment: list of subjects*]with the ability to verify evidence of the validity of the indicated information.



*This SFR describes the output of the Prove service provided by the DSC. The evidence of validity and/or authenticity, or other evidence derived, is expected to be processed by the Root of Trust for Measurement. Additionally, the use of a Root of Trust for Reporting presupposes a logging capability or other means of generating state information that could be conveyed to external entities. Therefore, FDP\_DAU.1.1/Prove must be selected if-and-only-if the Root of Trust for Measurement and the Root of Trust for Reporting are both selected in FPT\_ROT\_EXT.1.1. An ‘authenticated user’ in the sense of the selection in FDP\_DAU.1.1/Prove means a user who has been authenticated by the DSC according to the requirements of FIA\_UAU.5/Prove. Therefore, FIA\_UAU.5/Prove must be included if-and-only-if “authenticated user” is selected in FDP\_DAU.1.1/Prove.*

*In FDP\_DAU.1.1/Prove, the DSC will issue a validity-stamped or authenticity-stamped piece of data. In this case, validity-stamped means that the form of the issued data enables an external entity to verify that the data has been issued via the DSC’s Prove service. The implementation might be via a DSC cryptographic signature, or a MAC using a symmetric key shared with the receiver, for example. Authenticity-stamped means that the DSC can verify that the user providing this data for any Root of Trust services that require generation of DSC state information is authentic based on the assumption that the user has been authenticated by the DSC.*

*Data that would need to be validity-stamped includes data over which the DSC is the authority, such as the state of its own firmware. Data that would need to be authenticity-stamped includes data about which the DSC knows nothing, but where it will issue the data with a statement that the DSC has authenticated the source of this data.*

*For data that is validity-stamped, the DSC does nothing but respond to a request to issue the data; thus, authentication of the user issuing the data is not needed and is covered by FDP\_DAU.1/Prove. Otherwise, in the case the DSC has no understanding of this data, a step is needed via FIA\_UAU.5/Prove by which the DSC authenticates the user for this service, and that DSC and/or Prove service will therefore vouch for the user, not the validity of the data itself.*

* + 1. FDP\_FRS\_EXT.2 Factory Reset Behavior

Not all DSCs permit a factory reset of the TOE, or perform a factory reset in response to excessive failed authentication attempts. Those that do are expected to perform a factory reset in a manner that prevents any inadvertent disclosure of security-relevant data that was present on the DSC prior to the factory reset. For DSCs that permit factory reset functionality (as indicated by selection of ‘factory reset the TOE wiping out all user SDOs, as described by FDP\_FRS\_EXT.2’ in FIA\_AFL\_EXT.1.3, or by ‘no actions or conditions’ NOT being selected in FDP\_FRS\_EXT.1.1), the following component must be included in the TOE boundary.

**FDP\_FRS\_EXT.2 Factory Reset Behavior**

**FDP\_FRS\_EXT.2.1** Upon initiation of a factory reset, the TSF shall destroy all user-specific SDOs and restore the following pre-installed SDOs to their factory settings:[*assignment: pre-installed SDOs to be restored during a factory reset*]*.*

* + 1. FDP\_MFW\_EXT.2 Basic Firmware Integrity

Data and firmware integrity is not a required component of this cPP in all cases because some DSCs will have immutable firmware. For DSCs where the firmware is mutable (as selected in FDP\_MFW\_EXT.1.1), the following component must be included in the TOE boundary.

**FDP\_MFW\_EXT.2 Basic Firmware Integrity**

**FDP\_MFW\_EXT.2.1** The TSF shall have the ability to verify the integrity of the firmware*.*

**FDP\_MFW\_EXT.2.2** The TSF shall provide a capability to generate evidence of the integrity of the firmware.

*The TOE guarantees the integrity of the firmware by verifying its integrity.*

*Verifying the integrity of the firmware could be accomplished by guaranteeing the validity of firmware within the TOE prior to execution.*

This requirement covers the case of ensuring the firmware is trustworthy in immutable form or mutable through any firmware updates, since the integrity and authenticity are checked prior to execution.

FCS\_COP.1/SigVer applies if the TOE provides the capability to update the TOE firmware and uses digital signatures and MAC verification for update verification. The ST Author should choose the algorithm implemented to perform digital signatures. For the algorithm(s) chosen, the ST author should make the appropriate assignments/selections to specify the parameters that are implemented for that algorithm.

* + 1. FDP\_MFW\_EXT.3 Firmware Authentication with Identity of Guarantor

Firmware authentication is not a required component of this cPP in all cases because some DSCs will have immutable firmware. For DSCs where the firmware is mutable (as selected in FDP\_MFW\_EXT.1.1), the following component must be included in the TOE boundary.

**FDP\_MFW\_EXT.3 Firmware Authentication with Identity of Guarantor**

**FDP\_MFW\_EXT.3.1** The TSF shall have the ability to verify the authenticity of the firmware*.*

**FDP\_MFW\_EXT.3.2** The TSF shall provide a capability to generate evidence of the authenticity of the firmware.

*The TOE guarantees the authenticity of the firmware by verifying its signature.*

*Verifying the authenticity of the firmware could be accomplished by guaranteeing the validity of firmware within the TOE prior to execution.*

This requirement covers the case of ensuring the firmware is trustworthy in immutable form or mutable through any firmware updates, since the integrity and authenticity are checked prior to execution.

FCS\_COP.1/SigVer applies if the TOE provides the capability to update the TOE firmware and uses digital signatures and MAC verification for update verification. The ST Author should choose the algorithm implemented to perform digital signatures. For the algorithm(s) chosen, the ST author should make the appropriate assignments/selections to specify the parameters that are implemented for that algorithm.

* 1. Identification and Authentication
     1. FIA\_AFL\_EXT.2 Authorization Failure Response

A DSC that conforms to this PP has many ways to protected against a brute-force attack on unauthorized SDO access, as defined by FIA\_AFL\_EXT.1.3. In some cases, subsequent access to the target data may be locked indefinitely until such time as an administrator re-authorizes access to it. For DSCs where the TSF’s response to a brute-force attack includes locking the data (as indicated by selection of ‘prevent all future authorization attempts indefinitely (i.e., lock)’ in FIA\_AFL\_EXT.1.3), the following component must be included in the TOE boundary.

**FIA\_AFL\_EXT.2 Authorization Failure Response**

**FIA\_AFL\_EXT.2.1** When the TSF locks an SDO (i.e. prevents authorization attempts for an SDO) due to a user exceeding the allowed threshold for unsuccessful authorization attempts, then only an administrator may unlock access to the SDO and reset the corresponding failed authorization attempt counter.

This SFR is applicable only when the TSF’s response to excessive authorization failures includes “prevent all future authorization attempts indefinitely (i.e., lock)” as specified by FIA\_AFL\_EXT.1.3.

* + 1. FIA\_UAU.5/Prove Multiple Authentication Mechanisms (Prove Service)

**FIA\_UAU.5/Prove Multiple Authentication Mechanisms (Prove Service)**

**FIA\_UAU.5.1/Prove**The TSF shall provide [*assignment: list of authentication mechanism(s)*] to support user authentication **for the Prove service**.

**FIA\_UAU.5.2/Prove**The TSF shall authenticate any user’s claimed identity according to the [*assignment: rules describing how each authentication mechanism provides authentication for the Prove service*].



*This SFR describes the authentication required from a user of the Prove service as a precondition for providing an output as defined by the selection in FDP\_DAU.1.1/Prove where the relevant data is declared valid by an authenticated user. Therefore, FIA\_UAU.5/Prove must be included in an ST if-and-only-if the “authenticated user” selection is declared in the ST in FDP\_DAU.1.1/Prove (from the selection-based requirements section).*

* 1. Protection of the TSF
     1. FPT\_FLS.1/FW Failure with Preservation of Secure State (Firmware)

A DSC’s ability to handle failures related to authenticity, integrity, and invalid versions of firmware is not applicable in all cases because some DSCs will have immutable firmware. For DSCs where the firmware is mutable (as selected in FDP\_MFW\_EXT.1.1), the following component must be included in the TOE boundary.

**FPT\_FLS.1/FW Failure with Preservation of Secure State (Firmware)**

**FPT\_FLS.1.1/FW** The TSF shall preserve a secure state when the following types of **firmware** failures occur: [*authenticity violation, integrity violation, rollback violation*].

*The phrase “secure state” refers to a state in which the TOE has consistent TSF data and a TSF that can correctly enforce the policy. The TOE must ensure that no further processing of TSF or user data takes place while in an insecure state. This state may be the initial “boot” of a clean system, or it might be some check-pointed state. It is expected that in most cases, the TOE will halt and require a reset or reinitialization to return to a known secure state.*

* + 1. FPT\_ITT.1 Basic Internal TSF Data Transfer Protection

FPT\_ITT.1 Basic Internal TSF Data Transfer Protection

**FPT\_ITT.1.1** The TSF shall protect TSF data from [disclosure] **and [selection: modification, no other actions]** when it is transmitted between separate parts of the TOE.

* + 1. FPT\_RPL.1/Rollback Replay Detection (Rollback)

A DSC’s ability to detect an attempted rollback (software/firmware downgrade) is not applicable in all cases because some DSCs will have immutable firmware that cannot be modified in any way. For DSCs where the firmware is mutable (as selected in FDP\_MFW\_EXT.1.1), the following component must be included in the TOE boundary.

**FPT\_RPL.1/Rollback Replay Detection (Rollback)**

**FPT\_RPL.1.1/Rollback** The TSF shall detect replay for the following entities: [*previous firmware builds*].

**FPT\_RPL.1.2/Rollback** The TSF shall **prevent the execution of the loaded firmware and perform [selection, choose one of: [*assignment: other actions*], no other actions]** when replay is detected.

*The TSF data is used as a guarantee of the ordinal identifier of the firmware instance. When a firmware load is requested, the TSF ensures the authenticated firmware ordinal identifier is greater than or equal to the previously authenticated firmware identifier. For example, this could be accomplished by ensuring the validated instance of the firmware to be loaded is greater than or equal to the instance previously validated and loaded into the TOE. By loading a previous instance of firmware, it potentially opens up the device to known vulnerabilities.*

* 1. Trusted Path/Channels
     1. FTP\_CCMP\_EXT.1 CCM Protocol

FTP\_CCMP\_EXT.1 CCM Protocol

**FTP\_CCMP\_EXT.1.1** The TSF shall implement CCMP (IEEE 802.11i) using [selection: AES, Camellia] in CCM mode and key size [selection: 128-bits, 256-bits].

**FTP\_CCMP\_EXT.1.2** The TSF shall discard incoming messages if authentication fails.

**FTP\_CCMP\_EXT.1.3** The TSF shall discard incoming messages that are malformed or invalid.

*Inclusion of this SFR requires inclusion of AES-CCM and/or CAM-CCM in FCS\_COP.1/SKC.*

* + 1. FTP\_GCMP\_EXT.1 GCM Protocol

FTP\_GCMP\_EXT.1 GCM Mode Protocol

**FTP\_GCMP\_EXT.1.1** The TSF shall implement GCMP (IEEE 802.11ad) using [selection: AES, Camellia] in GCM mode and key size [selection: 128-bits, 256-bits].

**FTP\_GCMP\_EXT.1.2** The TSF shall discard incoming messages if authentication fails.

**FTP\_GCMP\_EXT.1.3** The TSF shall discard incoming messages that are malformed or invalid.

Inclusion of this SFR requires inclusion of AES-GCM and/or CAM-GCM in FCS\_COP.1/SKC.

* + 1. FTP\_ITC\_EXT.1 Cryptographically Protected Communications Channels

FTP\_ITC\_EXT.1 Cryptographically Protected Communications Channels

**FTP\_ITC\_EXT.1.1** The TSF shall use [selection: CCMP, GCMP] protocol to provide a communication channel between itself and [*assignment: list of IT entities external to the TOE*] that protects channel data from disclosure and ensures the integrity of channel data.

If CCMP is selected, the ST author must include FTP\_CCMP\_EXT.1.

If GCMP is selected, the ST author must include FTP\_GCMP\_EXT.1.

* + 1. FTP\_ITE\_EXT.1 Encrypted Data Communications

FTP\_ITE\_EXT.1 Encrypted Data Communications

**FTP\_ITE\_EXT.1.1** The TSF shall encrypt data for transfer between the TOE and [*assignment: list of IT entities external to the TOE*] using a cryptographic algorithm and key size as specified in FCS\_COP.1/SKC, and using [selection:

* Pre-shared keys;
* Keys established according to FCS\_CKM.2;
* Keys exchanged using a physically protected communication mechanism conformant with FTP\_ITP\_EXT.1

].

*This requirement applies to encrypted data communications between the TOE and external entities that do not use a physically protected mechanism conforming to FTP\_ITP\_EXT.1, or a cryptographically protected data channel as conforming to FTP\_ITC\_EXT.1. For example, if data is transferred through encrypted buffers (or blobs) then this requirement applies. If data is transferred through a physically protected channel, then FTP\_ITP\_EXT.1 applies.*

* + 1. FTP\_ITP\_EXT.1 Physically Protected Channel

FTP\_ITP\_EXT.1 Physically Protected Channel

**FTP\_ITP\_EXT.1.1** The TSF shall provide a **physically protected** communication channel between itself and [*assignment: list of other IT entities within the same platform*].

1. Extended Component Definitions

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

(Note: formatting conventions for selections and assignments in this Appendix are those in [CC2].)

1. Entropy Documentation and Assessment

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

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

* 1. Design Description

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

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

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

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

* 1. Entropy Justification

There should be a technical argument for where the unpredictability in the source comes from and why there is confidence in the entropy source exhibiting probabilistic behavior (an explanation of the probability distribution and justification for that distribution given the particular source is one way to describe this). This argument will include a description of the expected entropy rate and explain how you ensure that sufficient entropy is going into the TOE randomizer seeding process. This discussion will be part of a justification for why the entropy source can be relied upon to produce bits with entropy.

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

* 1. Operating Conditions

Documentation will also include the range of operating conditions under which the entropy source is expected to generate random data. It will clearly describe the measures that have been taken in the system design to ensure the entropy source continues to operate under those conditions. Similarly, documentation shall describe the conditions under which the entropy source is known to malfunction or become inconsistent. Methods used to detect failure or degradation of the source shall be included.

* 1. Health Testing

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

1. Glossary

| Term | Meaning |
| --- | --- |
| Access | In the context of SDOs, access to an SDO represents the list of actions permissible with an SDO, including its generation, use, modification, propagation, and destruction. |
| Attestation | The process of presenting verifiable evidence describing those characteristics that affect integrity. Examples of these characteristics are boot firmware and boot critical data which, combined, describe the way the DSC booted. [SA] |
| Attributes | Indications of characteristics or properties of the SDEs bound in an SDO. |
| Authorization Value | Critical data bound to an action by itself or to action on a subject. Such data, when presented to the TOE, authorizes the action by itself or authorizes the action on or with the subject respectively. |
| Authentication Token | Critical data bound to a user. Such data, when presented to the TOE and successfully verified by it, authenticates the user. The TOE may use the successful authentication of a user as an authorization to execute an action on its behalf. |
| Authenticator | A shortened name for Authentication Token. |
| Boot Critical Data | Critical data that persists across power cycles and determines characteristics of the DSC. Examples of boot critical data can be DSC configuration settings, certificates, and the results of measurements obtained by the root of trust for measurement. |
| Boot Firmware | The first firmware that executes during the boot process. |
| Chain of Trust | A Chain of Trust is anchored in a Root of Trust and extends a trust boundary by verifying the authenticity and integrity of successive components before passing control to those components. [SA] |
| Data Encryption Key | An encryption key, usually for a symmetric algorithm, that encrypts data that is not keying material. |
| Integrity | Assurance of trustworthiness and accuracy. |
| Immutable | Unchangeable. |
| Key Encryption Key | An encryption key that encrypts other keying material. This is sometimes called a key wrapping key. A KEK can be either symmetric or asymmetric. |
| Known Answer Tests (KATs) | Test vectors or data generated to determine the correctness of an implementation. |
| Operator | Human being who has physical possession of the platform on which the DSC is located. [GD] |
| Owner | Human being who controls/manages the platform on which the DSC is located. May be remote. [GD] |
| Platform | A platform consists of the hardware and firmware of a computing entity. |
| Pre-installed SDOs | An SDO installed on the DSC by the manufacturer. The SDO consists of an SDE and attributes, which if not explicitly expressed in a data structure, are implicit based on the functions that have exclusive access to the SDE. |
| Privileged Function | Functions restricted to the role of administrator, which may include, but are not limited to, provisioning keys, provisioning user authorization values, deprovisioning user authorization values, provisioning administrator authorization values, changing authorization values, disabling key escrow, and configuring cryptography. |
| Protected Data Blob | Data in an encrypted structure that protects its confidentiality and/or integrity (as required by the context in which it is used). |
| Protected Storage | Protected Storage usually refers to DSC hardware used to store SDEs or SDOs, and provide integrity protection for all items and confidentiality for those items that require it. Protected Storage may also refer to storage external to the DSC, which is usually encrypted by keys maintained by the DSC’s internal protected storage capabilities. |
| Protections | Mechanisms that ensure components of a DSC (executable firmware code and critical data) remain in a state of integrity and are protected from modification outside of authorized, authenticated processes and entities. [NIST-ROTM] |
| Remote Secure Channel | Logical channel to the DSC from a remote entity, which cryptographically protects the confidentiality and integrity of the channel content. |
| Root Encryption Key | An encryption key that serves as the anchor of a hierarchy of keys. |
| Root of Trust | A root of trust performs one or more security specific functions; establishing the foundation on which all trust in a system is placed. [NIST-ROTM] |
| Root of Trust for Measurement | A root of trust responsible for measuring the boot firmware and critical configuration data, which serve as indicators for the way the DSC, booted. |
| Root of Trust for Reporting | A root of trust responsible for reporting the results of the measurements obtained by the root of trust for measurement. |
| Root of Trust for Storage | A root of trust responsible for storing, and preserving the integrity of, boot critical data. |
| Root of Trust for Update | A root of trust responsible for updating the firmware. |
| Root of Trust for Verification | A root of trust responsible for verifying digital signatures. |
| Security Data Element | A Critical Security Parameter, such as a cryptographic key or authorization token. |
| Security Data Object | A Security Data Object (SDO) may include one or more SDEs. SDOs bind SDE(s) with a set of attributes. |
| Symmetric Encryption Key | A value intend to input as a key to a symmetric encryption algorithm, such as AES. |
| System | A system consists of the platform hardware and firmware in addition to the higher-level software running on top of it (kernel, user-space processes, etc.). |
| Trusted Local Channel | Physical channel to the DSC within the platform of which the DSC is a part, which is protected by the operational environment to ensure confidentiality and integrity. |
| User | Entity who relies on the services provided by the platform and/or DSC. |

Table 16: Glossary

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

1. Acronyms

| Acronym | Meaning |
| --- | --- |
| AEAD | Authenticated Encryption with Associated Data |
| AES | Advanced Encryption Standard |
| ANSI | American National Standards Institute |
| CA | Certificate Authority |
| CBC | Cipher Block Chaining |
| CCM | Counter with CBC-Message Authentication Code |
| CCMP | CCM Protocol |
| CMC | Certificate Management over Cryptographic Message Syntax (CMS) |
| CPU | Central Processing Unit |
| CRL | Certificate Revocation List |
| CSP | Critical Security Parameters |
| DAR | Data At Rest |
| DEK | Data Encryption Key |
| DH | Diffie-Hellman |
| DSA | Digital Signature Algorithm |
| ECDH | Elliptic Curve Diffie Hellman |
| ECDSA | Elliptic Curve Digital Signature Algorithm |
| EEPROM | Electrically Erasable Programmable Read-Only Memory |
| FIPS | Federal Information Processing Standards |
| FQDN | Fully Qualified Domain Name |
| GCM | Galois Counter Mode |
| HMAC | Keyed-Hash Message Authentication Code |
| HTTPS | Hypertext Transfer Protocol Secure |
| IEEE | Institute of Electrical and Electronics Engineers |
| IP | Internet Protocol |
| IPSec | Internet Protocol Security |
| KEK | Key Encryption Key |
| NIST | National Institute of Standards and Technology |
| OCSP | Online Certificate Status Protocol |
| OS | Operating System |
| PBKDF | Password-Based Key Derivation Function |
| PMK | Pairwise Master Key |
| PP | Protection Profile |
| PTK | Pairwise Temporal Key |
| RA | Registration Authority |
| RBG | Random Bit Generator |
| REK | Root Encryption Key |
| ROM | Read-only memory |
| RSA | Rivest Shamir Adleman Algorithm |
| SDE | Security Data Element |
| SDO | Security Data Object |
| SFP | Security Function Policy |
| SFR | Security Functional Requirement |
| SHA | Secure Hash Algorithm |
| SK | Symmetric Key or Symmetric Encryption Key |
| SPI | Security Parameter Index |
| SSH | Secure Shell |
| ST | Security Target |
| TLS | Transport Layer Security |
| TOE | Target of Evaluation |
| TSF | TOE Security Functionality |
| TSS | TOE Summary Specification |
| USB | Universal Serial Bus |
| USSD | Unstructured Supplementary Service Data |

Table 17: Acronyms

1. References

[NIST-DRBG] Barker, Elaine, and John Kelsey. *National Institute of Standards and Technology (NIST) Special Publication 800-90A, Recommendation for Random Number Generation Using Deterministic Random Bit Generators*, National Institute of Standards and Technology, January 2012

[NIST-RSA] Barker, Elaine, Lily Chen, Andrew Regenscheid, and Miles Smid. *National Institute of Standards and Technology (NIST) Special Publication 800-56B, Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography*, National Institute of Standards and Technology, August 2009.

[NIST-CMAC] Dworkin, Morris. *National Institute of Standards and Technology (NIST) Special Publication 800-38B, Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*, National Institute of Standards and Technology, December 2012.

[GD] Grawrock, David. Dynamics of a Trusted Platform: A building block approach. Intel Press, 2009.

[ISO-HASH] ISO/IEC. ISO/IEC 10118-3:2018 *IT Security Techniques – Hash-Functions – Part 3: Dedicated Hash-Functions*, ISO/IEC, October 2018

[ISO-CIPH] ISO/IEC. *ISO/IEC 18033-3:2010 Information Technology – Security Techniques – Encryption Algorithms – Part 3: Block Ciphers*, ISO/IEC, December 2012

[ISO-CMAC] ISO/IEC. *ISO/IEC 9797-1 Information Technology – Security Techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms Using a Block Cipher,* ISO/IEC, December 1999

[ISO-HMAC] ISO/IEC. *ISO/IEC 9797-2:2011 Information Technology – Security Techniques – Message Authentication Codes (MACs) – Part 2: Mechanisms Using a Dedicated Hash-Function*

[FIPS-SHA] National Institute of Standards and Technology. *Federal Information Processing Standard Publication (FIPS-PUB) 180-4, Secure Hash Standard (SHS)*, National Institute of Standards and Technology, March 2012.

[FIPS-HMAC] National Institute of Standards and Technology. *Federal Information Processing Standard Publication (FIPS-PUB) 198-1, The Keyed-Hash Message Authentication Code (HMAC)*, National Institute of Standards and Technology, July 2008.

[SA] Segall, Ariel. *Trusted Platform Modules: Why, When and How to Use Them*. The Institution of Engineering and Technology, 2016.

[NIST-KDRV] Barker, Elaine, Lily Chen, and Rich Davis. *NIST Special Publication 800-56C, Recommendation for Key-Derivation Methods in Key-Establishment Schemes*, National Institute of Standards and Technology, April 2018.

[NIST-KDV] Kelsey, John, Shu-jen Chang, and Ray Perlner*. NIST Special Publication 800-185 SHA-3 Derived Functions: cSHAKE, KMAC, TupleHash, and ParallelHash*, National Institute of Standards and Technology, December 2016.

[NIST-ROTM] Chen, Lily, Joshua Franklin, and Andrew Regenscheid. *National Institute of Standards and Technology (NIST) Special Publication 800-164 (Draft), Guidelines on Hardware Rooted Security in Mobile Devices (Draft)*, National Institute of Standards and Technology, October 2012.

1. For details see <http://www.commoncriteriaportal.org/> [↑](#footnote-ref-2)
2. In certain cases, SHA-1 may also be used for verifying old digital signatures and time stamps, provided that this is explicitly allowed by the application domain. [↑](#footnote-ref-3)
3. re-authenticate [↑](#footnote-ref-4)