TuffServ® Intelligent E-2 Common Data Loader (iECDL) v2.74.1 Security Target

Version 1.20

07 January 2026

Ampex Inc.
26450 Corporate Ave
Suite 200
Hayward, CA 94545-3914

2400 Research Blvd
Suite 395
Rockville, MD 20850

Revision History

Version	Date	Changes
Version 1.6	21 November 2024	Draft
Version 1.7	25 November 2025	Separated the signature verification keys.
Version 1.8	16 September 2025	Addressed check-in ECRs.
Version 1.10	07 November 2025	Updated based on evaluator comments.
Version 1.11	07 November 2025	Updated based on evaluator comments.
Version 1.15	29 November 2025	Updated based on vendor mods to the code.
Version 1.16	29 November 2025	Frozen for evaluation.
Version 1.17	30 November 2025	Minor mods as the result of completing the KMD.
Version 1.18	08 December 2025	Minor mods.
Version 1.19	02 January 2026	Addressed ECRs.
Version 1.20	07 January 2026	Addressed ECRs.

Table of Contents

List of Figures

List of Tables

1 Introduction

The Security Target (ST) serves as the basis for the Common Criteria (CC) evaluation and identifies the Target of Evaluation (TOE), the scope of the evaluation, and the assumptions made throughout. This document will also describe the intended operational environment of the TOE, and the security functional and security assurance requirements that the TOE meets.

1.1 Security Target and TOE Reference

This section provides the information needed to identify and control the TOE and the ST.

Table 1: TOE/ST Identification

Category	Identifier
ST Title	TuffServ® Intelligent E-2 Common Data Loader (iECDL) v2.74.1 Security Target
ST Version	1.20
ST Date	07 January 2026
ST Author	Acumen Security
TOE Identifier	TuffServ® Intelligent E-2 Common Data Loader (iECDL)
TOE Version	2.74.1
TOE Developer	Ampex Data Systems Corporation
Key Words	Encryption Engine, Full Disc encryption

1.2 Product Overview

This Security Target is for the Common Criteria evaluation of the Ampex Data Systems Corporation TuffServ® Intelligent E-2 Common Data Loader (iECDL) version 2.74.1 against the *collaborative Protection Profile for Full Drive Encryption – Authorization Acquisition*, Version 2.0E and the *collaborative Protection Profile for Full Drive Encryption – Encryption Engine*, Version 2.0E. The iECDL is a hardware full drive encryption device that implements the selections required by the *Hardware Full Drive Encryption CSfC Components List*.

The Ampex iECDL is an intelligent ground download station that provides the tools for managing the removable storage media utilized by a military on board vessel's network file server (NFS). As a data loader, the iECDL provides the capability to store and retrieve digital data from a classified nonvolatile solid-state transportable Removable Storage media (RSM) or an intelligent Removable Storage Media (iRSM). The data includes preflight loading of Operation Flight Program (OFP), intelligence data, maps, tactical programs, and other mission related data via a workstation using a 5Gb Ethernet.

The iECDL implements a Key Transfer Device (KTD) for secure storage of cryptographic keys as well as a secure mechanism to assign and distribute cryptographic keys for the removable media distributed across the fleet. The iECDL supports multiple protocols which may be used concurrently, including File Network File System (NFS), Secure Copy Protocol (SCP), Trivial File Transfer Protocol (TFTP), and Secure Shell (SSH).

1.3 Usage and Major Security Features of the TOE

The TOE conforms to the *collaborative Protection Profile for Full Drive Encryption – Authorization Acquisition*, Version 2.0 + Errata (FDE_AA cPP) and *collaborative Protection Profile for Full Drive Encryption – Encryption Engine*, Version 2.0 + Errata (FDE_EE cPP).

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

1.4 TOE Type

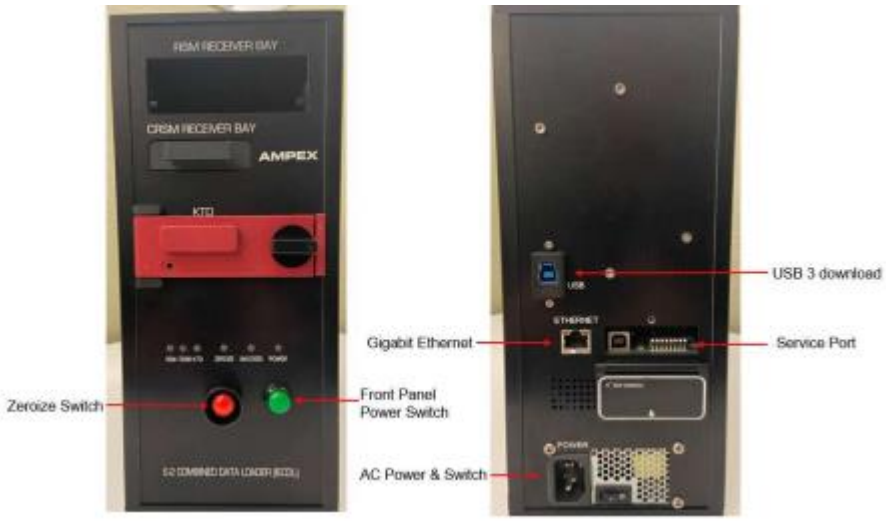
The TOE is a hybrid full disc encryption solution referred to in FDE_AA and FDE_EE cPPs. The TOE is a hardware, firmware, and software encryption and decryption, key management, and policy enforcement product. The TOE implements a full set of functions required for the AA and EE components.

1.5 TOE Description

The TOE is an Authorization Acquisition (AA) module and Encryption Engine (EE) module for the Ampex TuffServ® Full Drive Encryption (FDE) solution protecting sensitive data at rest. It implements the data encryption, policy enforcement, and key management functions of the file server. User data is encrypted by the TOE prior to being forwarded to the attached drives for storage and decrypted when retrieved from the drives. Both the encryption and the decryption occur without user intervention and cannot be disabled.

Communicating with the iECDL for management and control can be done remotely via an RJ45 connector used for system control via Ethernet. In addition, the iECDL can be controlled directly via serial port connection. The iECDL supports two management applications: the iECDL Graphical User Interface (GUI) and iECDL Command Line Interface (CLI). The LAN (Local Area Network) provides the GUI interface and the serial port provides the CLI interface.

Figure 1: Ampex Data Systems TuffServ® Intelligent E-2 Common Data Loader (iECDL) v2.74.1



The iECDL supports the following four external storage devices. Two, the Key Transfer Device (KTD) and the cRSM (compact Removable Storage Media) are used for input of the authentication factor. The Authentication Factor determines if iECDL User and Administrator can access the protected data. The other two, the Removable Storage Media (RSM) and the intelligent Removable Storage Media (iRSM)[1] are used to hold the data that is being encrypted. The four devices are described below.

An iECDL contains two CPU (Central Processing Unit) cards: the Host System Card runs the Ampex Computing Platform (ACCE) software. The ACCE provides the Full Drive Encryption (FDE) Authorization Acquisition (AA) functionality. The second card, the TuffServ® Encryption Module (TSEM) provides the Encryption Engine (EE) FDE functionality.

The Host system provides the management interfaces to the GUI and the CLI, interacts with the two USB (Universal Serial Bus) devices that store the authentication factors that enable users to access the encrypted drives, and provides a bridge between the authentication storage devices and the TSEM. The TSEM is responsible for authenticating the authentication factors and encrypting and decrypting the user data. The two cards communicate with each over a USB connection.

AES-XTS (Advanced Encryption Standard- XEX (XOR Encrypt XOR) Tweakable Block Cipher with Ciphertext Stealing) is implemented on the FPGA (Field Programmable Gate Array) and uses DEKs (Data Encryption Key) derived from BEV (Border Encryption Value) for encrypting the user data stored on the attached drive and for decrypting it when retrieved. The FPGA is resident on the TSEM card.

The following section describes the four external storage devices.

compact Removable Storage Media (cRSM)
: The cRSM consists of a Solid State Drive (SSD) and a USB (Universal Serial Bus) connection to the iECDL. The cRSM holds the authorization factors for accessing protected data held on the RSMs/iRSMs.

Key Transfer Device (KTD)
: Mechanically, the KTD is identical to the standard cRSM, but cosmetically anodized in a different color. The KTD devices serve as a repository of the complete set of all authentication factors (auth factor) in the system. It is expected that they will be stored in locked containers when not in use and removed only when required to update auth factors installed on a cRSM.

The iECDL supports two USB ports for authentication input. The system will support one cRSM inserted or one KTD inserted, or both inserted.

Removable Storage Media (RSM) and intelligent Removable Storage Media (iRSM)[2]
: The RSM and iRSM are an array of SATA (Serial Advanced Technology Attachment) solid-state drives (SSD). The RSM and iRSM are used for storing the user data protected by the TOE. The RSM/iRSM includes a small amount of EEPROM (Electrically Erasable Programmable Read-Only Memory) for storing configuration and secondary security elements for key management.

1.5.1 Evaluated Configuration

The iECDL is configured with an RSM as the storage media, a KTD and a cRSM as the input devices for the authentication factors.

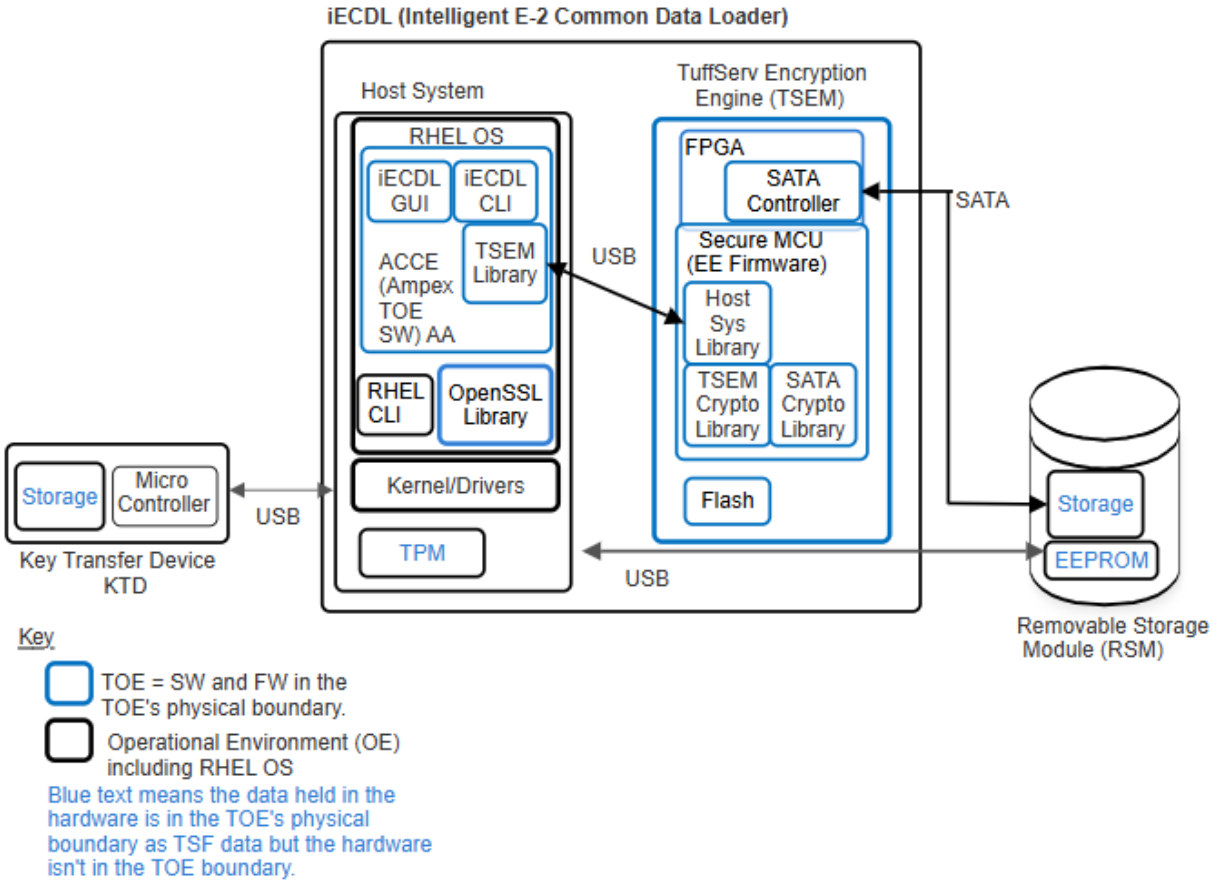
[1] Refer to Appendix A for a description of the differences between an RSM and an iRSM.

[2] Refer to Appendix A for a description of the differences between an RSM and an iRSM.

1.5.2 Physical Boundary

The following figure illustrates a typical iECDL with a KTD and an RSM.

Figure 2: iECDL TOE Components



The following section provides a description of the iECDL components and identifies which items are included in the TOE's physical boundary. Items included in the TOE physical boundary are identified in blue **font**. Items in the Operational Environment are identified with *italicized text*. Refer to Figure 2 for additional information.

iECDL

Hardware enclosure	The commercial grade casing hosting the TOE and other components of the iECDL. An iECDL includes two cards: a Host System card and the TSEM (TuffServ® Encryption Engine) card. Both are described in detail below.

Host System Card

Purpose	The Host System provides the Authorization Acquisition (AA) FDE functionality of the TOE. It is responsible for providing the interface to the TSEM and the Key Transfer Device (KTD)

	or compact Removable Storage Media (cRSM). The KTD and cRSM hold the authentication factors. Both provide the same functionality so, throughout the documentation for this evaluation, the documents refer to the KTD only. However, the description or functionality applies to both devices and both devices are included in the evaluated configuration. The iECDL requires a KTD or a cRSM to receive the authentication factor. The iECDL also support both inserted simultaneously. The iECDL will always select the authentication factor found on the KTD first. Additionally, the Host System provides the administrative interfaces that support management, control, and monitoring of the iECDL.
TOE Boundary	The TOE boundary on the Host System Card is the Ampex iECDL application, (the ACCE (Ampex Common Computing Environment)) and the AMPEX OpenSSL Cryptographic Module (the cryptographic library).
Processor/Microarchitecture	*Intel® Atom® C3558 (Goldmont microarchitecture)*
TPM	*Infineon SLB9665XT2.0 Trusted Platform Module* *The TPM provides the Host System's non-volatile memory. The **contents of the TPM** is TSF data and therefore included in the TOE's physical boundary.*
Ampex Software Build:	Ampex Common Computing Environment (ACCE), Version 2.74.1. The ACCE provides the AA (Authorization Acquisition) functionality. The ACCE is the interface between the KTD and the TSEM (TuffServ® Encryption Module) card. The ACCE provides two administrative interfaces: a GUI (Graphical User Interface) and a CLI (Command Line Interface) for iECDL management and control and includes the TSEM Library that provides the interface to the TSEM.
Cryptographic Library:	AMPEX OpenSSL Cryptographic Module, version 1.1.1, CAVP Cert #A7741 The iECDL uses the AMPEX OpenSSL Cryptographic Module (cryptographic library) to validate the signature of Host System software updates. There are no other cryptographic algorithms required by the ACCE.
Operating System	*RHEL 8.6* *The Operating System is included in the Operational Environment. iECDL Administrators interface to RHEL using the RHEL CLI in order to update the Host System software. Updating the ACCE software is done using RPM (Red Hat Package Manager). RPM calls the AMPEX OpenSSL Cryptographic Module (cryptographic library), version 1.1.1 to validate the upgrade digital signature.*

OS Kernel: Version 4.18-372

TuffServ® Encryption Module (TSEM)

Purpose	The TSEM provides the Encryption Engine (EE) FDE functionality. The TSEM provides the interface between the Host System card and the storage devices, the RSMs/iRSMS The TSEM provides cryptographic key management services for the TuffServ® secure storage device.
TOE Boundary	The complete TSEM card is included in the TOE boundary. This includes all of the hardware and software identified below.
TSEM Hardware	Version v1320249-010 Rev A
Processor/Microarchitecture:	NXP Kinetis K81 32-bit Security Microcontroller (Secure MCU) (with microarchitecture ARM Cortex M4)
FPGA (Field Programmable Gate Array)	The SATA (Serial Advanced Technology Attachment) Controllers are the IntelliProp Lycan FPGAs which implement all data plane security functions of the TOE. Data plane security functions include AES-XTS encryption and decryption of all data to and from the storage media (RSM/iRSM). The SATA Controller registers store Data Encryption Key (DEK) that encrypts and decrypts the data stored in the RSMs.
TSEM Firmware	Version iECDL_FW_1.1.16 The SoC implements control plane functionality, such as module initialization, configuration, and provisioning. The TSEM communicates with the Host System over a USB interface. The TSEM FW includes a Host System Library that provides the routines to interface to the Host System Card. **Cryptographic Libraries** The TSEM firmware includes the following two cryptographic libraries. **TSEM Cryptographic Library** v1.1.16.0, CAVP (Cryptographic Algorithm Verification Program) Cert #A2921 The TSEM Cryptographic Library provides the cryptographic primitives required for the TSEM EE firmware. **IPC-BL120B-ZM Library version EP2AGX65 (SATA Crypto Library)**, CAVP Cert #A2914 The IntelliProp Lycan FPGA's AES-XTS Encryption IP core. Used in the IntelliProp Lycan FPGA SATA controllers to encrypt and decrypt data stored on the RSM.

Key Transfer Device (KTD)

The KTD is in the Operational Environment. However, the authentication factors stored on the KTD are included in the TOE boundary (identified as KTD Storage in Figure 2). The KTD store the AA authorization factors used to determine if the encrypted/decrypted RSM data can be accessed.

Removable Storage Media (RSM)

The RSM/iRSM is in the Operational Environment (described in Section 1.5). However, the encrypted User data (identified as RSM Storage in Figure 2) stored on the RSM and the data stored in EEPROM (Electrically Erasable Programmable Random Access Memory) (identified as RSM EEPROM in Figure 2) are included in the TOE boundary.

1.5.2.1 TOE Operational Environment

The TOE's environmental components identified in the previous section in italicized text are required to operate the TOE in the evaluated configuration.

Additionally, a workstation is required to manage the iECDL. The workstation must be connected on the same LAN as the iECDL. The workstation must have the latest versions of Chrome or Edge browser installed.

1.5.3 Security Functions Provided by the TOE

The TOE provides the security functions required by the FDE_AA cPP and FDE_EE cPP.

1.5.3.1 Cryptographic Support

The TOE includes NIST CAVP-validated cryptographic algorithms supporting cryptographic functions. The TOE provides Key Derivation, BEV Validation, and data encryption.

1.5.3.2 User Data Protection

The TOE performs Full Drive Encryption such that the drive contains no plaintext user data. The TOE performs user data encryption by default in the out-of-the-box configuration using AES-XTS-256 mode.

1.5.3.3 Security Management

The TOE supports management functions for changing and erasing the DEK using the iECDL GUI (Graphical User Interface) and initiating the TOE updates using the iECDL command line interface (CLI).

1.5.3.4 Protection of the TSF

The TOE provides trusted firmware updates, protects Key and Key Material; and supports power saving states. The TOE runs a suite of self-tests during initial start-up (on power on).

1.5.4 TOE Documentation

The following documents are essential to understanding and controlling the TOE in the evaluated configuration and included in the TOE physical boundary.

- *TuffServ® Intelligent E-2 Common Data Loader (iECDL) v2.74.1 Common Criteria User Guide,* Version 1.5, January 2026 (AGD_CC).

- *TuffServ® Intelligent E-2 Common Data Loader (iECDL)* v2.74.1 *Security Target,* Version 1.20, 07 January 2026.

1.6 Product Functionality not Included in the Scope of the Evaluation

The following functionality is not in the scope of the evaluation:

- LEDs (Light Emitting Diode) on the front panel of the TOE are not included in the evaluation activities.
- The network interface used by the TOE Users and the supported network application protocols that access the iECDL memory (File Network File System (NFS), Secure Copy Protocol (SCP), Trivial File Transfer Protocol (TFTP), and Secure Shell (SSH)) are not included in the evaluated configuration.
- When the iECDL boots, there is a packet exchange between the KTD (Key Transfer Device) and the TPM (Trusted Platform Module). The purpose is to "trust" the KTD. This exchange or level of security is not included in the evaluation. All KTDs are considered trusted.
- Users (non-Administrator) access the TOE's protected data via File Network File System (NFS), Secure Copy Protocol (SCP), Trivial File Transfer Protocol (TFTP), and Secure Shell (SSH). The TOE supports the roles provided and managed by the RHEL Operating System (Operational Environment), allowing Users with privilege (superuser or root) access to files and applications stored on the protected disk, whereas non-privileged users will be denied access. This level of user roles, that determine which users have access to which data, is not included in the evaluated configuration.
- The iECDL provides an additional layer of security on top of the RHEL role based security implemented. Upon login, the iECDL insures that the User is allowed to invoke the specific command. This upper-level of security is not included in the evaluation.
- Cryptographic Officer (CO)

1.7 Security Target Conventions

- The iECDLs support a KTD (Key Transfer Device) and a cRSM (compact Removable Storage Media) as authentication factor input devices. Their detailed description is in Section 1.5 TOE Description of this document. Both provide the same functionality so, throughout the documentation for this evaluation, the documents refer to the KTD only. However, the description or functionality applies to both devices and both devices are included in the evaluated configuration.
- The iECDLs include an RSM (Removable Storage Media) or an iRSM (intelligent Removable Storage Media) as storage drives. Their detailed description is in Section 1.5 TOE Description of this document. Both the RSM and the iRSM provide the same security functionality so, throughout the documentation for this evaluation, the documents refer to a RSM only as a storage drive. However, the description or functionality applies to both the RSM and iRSM.

2 Conformance Claims

This section identifies the TOE conformance claims, conformance rational, and relevant Technical Decisions (TDs).

2.1 CC Conformance Claims

The TOE is Common Criteria Part 2 extended and Common Criteria Part 3 conformant. The exact versions of each are the following:

- Common Criteria for Information Technology Security Evaluation Part 2, Version 3.1, Revision 5, April 2017, Extended.
- Common Criteria for Information Technology Security Evaluation Part 3, Version 3.1, Revision 5, April 2017, Conformant.

2.2 Protection Profile Conformance

This ST claims exact conformance to the following PP-Configuration:

PP-Configuration for Full Drive Encryption – Authorization Acquisition and Full Drive Encryption – Encryption Engine, Version: 1.0 (CFG_CPP_FDE_AA-CPP_FDE_EE_V1.0).

The PP-Configuration includes the following components:

- Base-PP: *collaborative Protection Profile for Full Drive Encryption – Authorization Acquisition*, Version 2.0 + Errata 20190201, February 1, 2019 *(FDE_AA)*.
- Base-PP: *collaborative Protection Profile for Full Drive Encryption – Encryption Engine*, Version 2.0 + Errata 20190201, February 1, 2019 *(FDE_EE)*.

2.3 Conformance Rationale

This ST provides exact conformance to FDE_EE v2.0 and exact conformance to FDE_AA v2.0. The security problem definition, security objectives, and security requirements in this ST are all taken from the Protection Profiles, performing only the operations defined therein.

2.4 Technical Decisions

Each NIAP Technical Decisions (TD) issued to date and applicable to the FDE EE v2.0 and FDE AA v2.0 cPP have been considered. The following table identifies all applicable TDs and states their applicability to the TOE.

Table 2: Technical Decisions

Relevant Technical Decision	cPP		Applicable to the Evaluation (Yes/No)	Notes and/or Exclusion Rationale (if applicable)
	FDE_AA	FDE_EE		
TD0929: FIT Technical Decision Clarification FCS_PCC_EXT.1.1.	✓		No	Archives TD0764. Archives TD0901. Modified FCS_PCC_EXT.1.1 SFR. The ST

Relevant Technical Decision	cPP		Applicable to the Evaluation (Yes/No)	Notes and/or Exclusion Rationale (if applicable)
	FDE_AA	FDE_EE		
				does not claim FCS_PCC_EXT.1.1.
TD0769: FIT Technical Decision for FPT_KYP_EXT.1.1	✓	✓	Yes	Applies to FPT_KYP_EXT.1/AA and FPT_KYP_EXT.1/EE SFRs.
TD0767: FIT Technical Decision for FMT_SMF.1.1 Refinement, found in Section 5.4 of CPE_FDE_AA_V2.0, and its associated test, found in Section 2.2.2.1.4 of CPE_FDE_AA_V2.0-SD	✓		Yes	Applies to SFR and Test for FMT_SMF.1.1/AA.
TD0766: FIT Technical Decision for FCS_CKM.4(d) test notes in Section 2.1.2.2.3 of CPP_FDE_AA_V2.0E and Section 4.1.1.5.3 of CPE_FDE_EE_V2.0E	✓	✓	Yes	Applies to Test only for FCS_CKM.4(d) in FCS_AA cPP and FCS_CKM.4(d) in FDE_EE cPP (not claimed).
TD0765: FIT Technical Decision for FMT_MOF.1 evaluation activities	✓		Yes	Applies to Test only for FMT_MOF.1.
TD0760: FIT Technical Decision for FCS_SNI_EXT.1.3 and FCS_COP.1(f)	✓		Yes	Applies to the SFR and TSS for FCS_SNI_EXT.1.3/AA and SFR for FCS_COP.1(f) (not claimed).
TD0759: FIT Technical Decision for FCS_AFA_EXT.1.1	✓		Yes	Applies to FCS_AFA_EXT.1 SFR.
TD606: FIT Technical Decision for FPT_KYP_EXT.1 evaluation activities	✓	✓	Yes	Does not affect any TOE evidence.
TD0464: FIT Technical Decision for FPT PWR EXT.1 compliant power savings states		✓	Yes	Applies to FPT_PWR_EXT.1/EE SFR.
TD0460: FIT Technical Decision for FPT PWR EXT.1 non-compliant power saving states		✓	Yes	Applies to AGD for FPT_PWR_EXT.1/EE.
TD0458: FIT Technical Decision for FPT_KYP_EXT.1 evaluation activities	✓	✓	Yes	Applies to TSS, AGD, and KMD for FPT_KYP_EXT.1/AA and FPT_KYP_EXT.1/EE.

3 Security Problem Definition

The security problem definition has been taken directly from the claimed PP and any relevant EPs/Modules/Packages specified in Section 2.2 and is reproduced here for the convenience of the reader. The security problem is described in terms of the threats that the TOE is expected to address, assumptions about the operational environment, and any Organizational Security Policies (OSPs) that the TOE is expected to enforce.

3.1 Threats

The threats included in the following table are drawn directly from the PP and any EPs/Modules/Packages specified in Section 2.2. The threats listed below that apply only to the FDE_AA or FDE_EE cPPs are appended with /AA or /EE respectively.

Table 3: Threats

ID	Threat
T.AUTHORIZATION_GUESSING/AA	Threat agents may exercise host software to repeatedly guess authorization factors, such as passwords and PINs. Successful guessing of the authorization factors may cause the TOE to release BEV or otherwise put it in a state in which it discloses protected data to unauthorized users.
T.AUTHORIZATION_GUESSING/EE	Threat agents may exercise host software to repeatedly guess authorization factors, such as passwords and PINs. Successful guessing of the authorization factors may cause the TOE to release DEKs or otherwise put it in a state in which it discloses protected data to unauthorized users.
T.CHOSEN_PLAINTEXT/EE	Threat agents may trick authorized users into storing chosen plaintext on the encrypted storage device in the form of an image, document, or some other file. A poor choice of encryption algorithms, encryption modes, and initialization vectors along with the chosen plaintext could allow attackers to recover the effective DEK, thus providing unauthorized access to the previously unknown plaintext on the storage device.
T.KEYING_MATERIAL_COMPROMISE/AA	Possession of any of the keys, authorization factors, submasks, and random numbers or any other values that contribute to the creation of keys or authorization factors could allow an unauthorized user to defeat the encryption. The cPP considers possession of key material of equal importance to the data itself. Threat agents may look for key material in unencrypted sectors of the storage device and on other peripherals in the operating environment (OE), e.g. BIOS configuration, SPI flash.
T.KEYING_MATERIAL_COMPROMISE/EE	Possession of any of the keys, authorization factors, submasks, and random numbers or any other values that contribute to the creation of keys or authorization factors could allow an unauthorized user to defeat the encryption. The cPP considers possession of keying material of equal

ID	Threat
	importance to the data itself. Threat agents may look for keying material in unencrypted sectors of the storage device and on other peripherals in the operating environment (OE), e.g. BIOS configuration, SPI flash, or TPMs.
T.KEYSPACE_EXHAUST	Threat agents may perform a cryptographic exhaust against the key space. Poorly chosen encryption algorithms and/or parameters allow attackers to exhaust the key space through brute force and give them unauthorized access to the data.
T.KNOWN_PLAINTEXT/EE	Threat agents know plaintext in regions of storage devices, especially in uninitialized regions (all zeroes) as well as regions that contain well known software such as operating systems. A poor choice of encryption algorithms, encryption modes, and initialization vectors along with known plaintext could allow an attacker to recover the effective DEK, thus providing unauthorized access to the previously unknown plaintext on the storage device.
T.UNAUTHORIZED_DATA_ACCESS	The cPP addresses the primary threat of unauthorized disclosure of protected data stored on a storage device. If an adversary obtains a lost or stolen storage device (e.g., a storage device contained in a laptop or a portable external storage device), they may attempt to connect a targeted storage device to a host of which they have complete control and have raw access to the storage device (e.g., to specified disk sectors, to specified blocks).
T.UNAUTHORIZED_FIRMWARE_MODIFY/EE	An attacker attempts to modify the firmware in the SED via a command from the AA or from the host platform that may compromise the security features of the TOE.
T.UNAUTHORIZED_FIRMWARE_UPDATE/EE	An attacker attempts to replace the firmware on the SED via a command from the AA or from the host platform with a malicious firmware update that may compromise the security features of the TOE.
T.UNAUTHORIZED_UPDATE	Threat agents may attempt to perform an update of the product which compromises the security features of the TOE. Poorly chosen update protocols, signature generation and verification algorithms, and parameters may allow attackers to install software and/or firmware that bypasses the intended security features and provides them unauthorized access to data.

3.2 Assumptions

The assumptions included in Table 4 are drawn directly from the PPs and any relevant EPs/Modules/Packages. The assumptions listed below that apply only to the FDE_AA or FDE_EE cPPs are appended with /AA or /EE respectively.

Table 4: Assumption

ID	Assumption
A.INITIAL_DRIVE_STATE/AA	Users enable Full Drive Encryption on a newly provisioned or initialized storage device free of protected data in areas not targeted for encryption. The cPP does not intend to include requirements to find all the areas on storage devices that potentially contain protected data. In some cases, it may not be possible - for example, data contained in “bad” sectors. While inadvertent exposure to data contained in bad sectors or un-partitioned space is unlikely, one may use forensics tools to recover data from such areas of the storage device. Consequently, the cPP assumes bad sectors, un-partitioned space, and areas that must contain unencrypted code (e.g., MBR and AA/EE pre-authentication software) contain no protected data.
A.INITIAL_DRIVE_STATE/EE	Users enable Full Drive Encryption on a newly provisioned or initialized storage device free of protected data in areas not targeted for encryption. It is also assumed that data intended for protection should not be on the targeted storage media until after provisioning. The cPP does not intend to include requirements to find all the areas on storage devices that potentially contain protected data. In some cases, it may not be possible - for example, data contained in “bad” sectors. While inadvertent exposure to data contained in bad sectors or un-partitioned space is unlikely, one may use forensics tools to recover data from such areas of the storage device. Consequently, the cPP assumes bad sectors, un-partitioned space, and areas that must contain unencrypted code (e.g., MBR and AA/EE pre-authentication software) contain no protected data.
A.PASSWORD_STRENGTH/AA	Authorized administrators ensure password/passphrase authorization factors have sufficient strength and entropy to reflect the sensitivity of the data being protected.
A.PHYSICAL	The platform is assumed to be physically protected in its Operational Environment and not subject to physical attacks that compromise the security and/or interfere with the platform’s correct operation.
A.PLATFORM_I&A/AA	The product does not interfere with or change the normal platform identification and authentication functionality such as the operating system login. It may provide authorization factors to the operating system's login interface, but it will not change or degrade the functionality of the actual interface.
A.PLATFORM_STATE	The platform in which the storage device resides (or an external storage device is connected) is free of malware that could interfere with the correct operation of the product.

ID	Assumption
A.POWER_DOWN/AA	The user does not leave the platform and/or storage device unattended until all volatile memory is cleared after a power-off, so memory remnant attacks are infeasible. Authorized users do not leave the platform and/or storage device in a mode where sensitive information persists in non-volatile storage (e.g., lock screen). Users power the platform and/or storage device down or place it into a power managed state, such as a “hibernation mode”.
A.POWER_DOWN/EE	The user does not leave the platform and/or storage device unattended until all volatile memory is cleared after a power-off. This properly clears memories and locks down the device. Authorized users do not leave the platform and/or storage device in a mode where sensitive information persists in non-volatile storage (e.g., lock screen or sleep state). Users power the platform and/or storage device down or place it into a power managed state, such as a “hibernation mode”.
A.SECURE_STATE/AA	Upon the completion of proper provisioning, the drive is only assumed secure when in a powered off state up until it is powered on and receives initial authorization.
A.SINGLE_USE_ET/AA	External tokens that contain authorization factors are used for no other purpose than to store the external token authorization factors.
A.STRONG_CRYPTO	All cryptography implemented in the Operational Environment and used by the product meets the requirements listed in the cPP. This includes generation of external token authorization factors by a RBG.
A.TRAINED_USER/AA	Authorized users follow all provided user guidance, including keeping password/passphrases and external tokens securely stored separately from the storage device and/or platform.
A.TRAINED_USER/EE	Users follow the provided guidance for securing the TOE and authorization factors. This includes conformance with authorization factor strength, using external token authentication factors for no other purpose and ensuring external token authorization factors are securely stored separately from the storage device and/or platform. The user should also be trained on how to power off their system.
A.TRUSTED_CHANNEL	Communication among and between product components (e.g., AA and EE) is sufficiently protected to prevent information disclosure. In cases in which a single product fulfils both cPPs, then the communication between the components does not extend beyond the boundary of the TOE (e.g., communication path is within the TOE boundary). In cases in which independent products satisfy the requirements of the AA and EE, the physically close proximity of the two products during their operation means that the threat agent has very little opportunity to interpose itself in the channel between the two without the user noticing and taking appropriate actions.

3.3 Organizational Security Policies

There are no Organizational Security Policies applicable to the TOE.

4 Security Objectives

The security objectives have been taken directly from the claimed PPs and any relevant EPs/Modules/Packages and are reproduced here for the convenience of the reader.

4.1 Security Objectives for the TOE

There are no security objectives applicable to the TOE defined in the claimed PPs.

4.2 Security Objectives for the Operational Environment

Security objectives for the operational environment assist the TOE in correctly providing its security functionality. These objectives, which are found in the table below, track with the assumptions about the TOE operational environment. The threats listed below that apply only to the FDE_AA or FDE_EE cPPs are appended with /AA or /EE respectively.

Table 5: Security Objectives for the Operational Environment

ID	Security Objectives
OE.INITIAL_DRIVE_STATE	The OE provides a newly provisioned or initialized storage device free of protected data in areas not targeted for encryption.
OE.PASSPHRASE_STRENGTH	An authorized administrator will be responsible for ensuring that the passphrase authorization factor conforms to guidance from the Enterprise using the TOE.
OE.PHYSICAL	The Operational Environment will provide a secure physical computing space such than an adversary is not able to make modifications to the environment or to the TOE itself.
OE.PLATFORM_I&A/AA	The Operational Environment will provide individual user identification and authentication mechanisms that operate independently of the authorization factors used by the TOE.
OE.PLATFORM_STATE/AA	The platform in which the storage device resides (or an external storage device is connected) is free of malware that could interfere with the correct operation of the product.
OE.POWER_DOWN/AA	Volatile memory is cleared after power-off so memory remnant attacks are infeasible.
OE.POWER_DOWN/EE	Volatile memory is cleared after entering a Compliant power saving state or turned off so memory remnant attacks are infeasible.
OE.SINGLE_USE_ET	External tokens that contain authorization factors will be used for no other purpose than to store the external token authorization factor.
OE.STRONG_ENVIRONMENT_CRYPTO	The Operating Environment will provide a cryptographic function capability that is commensurate with the requirements and capabilities of the TOE and Appendix A.
OE.TRAINED_USERS	Authorized users will be properly trained and follow all guidance for securing the TOE and authorization factors.
OE.TRUSTED_CHANNEL	Communication among and between product components (i.e., AA

ID	Security Objectives
	and EE) is sufficiently protected to prevent information disclosure.

5 Security Requirements

This section identifies the Security Functional Requirements (SFRs) and the Security Assurance Requirements (SARs) for the TOE. The Security Requirements included in this section are derived from Part 2 of the *Common Criteria for Information Technology Security Evaluation*, Version 3.1, Revision 5; the *collaborative Protection Profile for Full Drive Encryption – Authorization Acquisition*, Version 2.0 + Errata 20190201, and the *collaborative Protection Profile for Full Drive Encryption – Encryption Engine*, Version 2.0 + Errata 20190201.

5.1 Functional Requirements

5.1.1 Extended Functional Requirements

The extended requirements in this ST have been drawn from FDE_AA or FDE_EE v2.0 cPPs and are itemized below. The cPPs define the extended SFRs. Since they have not been redefined in this ST, the FDE_AA or FDE_EE cPP should be consulted for more information regarding these extensions to CC Part 2.

From FDE_AA cPP:

- FCS_AFA_EXT.1
- FCS_AFA_EXT.2
- FCS_CKM_EXT.4(a)
- FCS_CKM_EXT.4(b)/AA
- FCS_KYC_EXT.1
- FCS_SNI_EXT.1/AA
- FPT_KYP_EXT.1/AA
- FPT_PWR_EXT.1/AA
- FPT_PWR_EXT.2/AA
- FPT_TST_EXT.1/AA
- FPT_TUD_EXT.1/AA

From FDE_EE cPP:

- FCS_CKM_EXT.4(a)
- FCS_CKM_EXT.4(b)/EE
- FCS_CKM_EXT.6
- FCS_KDF_EXT.1
- FCS_KYC_EXT.2
- FCS_RBG_EXT.1
- FCS_SNI_EXT.1/EE
- FCS_VAL_EXT.1
- FDP_DSK_EXT.1
- FPT_FUA_EXT.1
- FPT_KYP_EXT.1/EE
- FPT_PWR_EXT.1/EE
- FPT_PWR_EXT.2/EE
- FPT_TST_EXT.1/EE
- FPT_TUD_EXT.1/EE

5.1.2 SFR Conventions

The CC allows the following types of operations to be performed on the functional requirements: assignments, selections, refinements, and iterations. The following font conventions are used within this document to identify operations defined by CC:

- Assignment: Indicated with *italicized* text;
- Refinement: Indicated with **bold** text;
- Selection: Indicated with underlined text;
- Iteration: The cPP authors indicate iterations within the cPPs by placing parenthesis placed around a lowercase letter following the SFR, e.g., FCS_COP.1(f).
  The ST author indicates iterations required for the identification of the source FDE_AA and FDE_EE cPP SFRs by placing a forward slash followed by either AA or EE after the SFR, e.g., FCS_CKM.4(a)/AA indicates that this SFR is from FDE_AA. The /AA or /EE iteration is applied:
  - if the SFR is included in both cPPs but the SFR text or operations are not identical;
  - if the SFR is included in both cPPs but the TSS, Operational Guidance, KMD, or Test assurance activities are not identical;
  - if the SFR is included in both cPPs and a TD applies to only one cPP;
  - if the SFR is included in both cPPs and the SFR includes a reference or selection to another SFR that has been iterated by the ST author; and
  - if the SFR is included in both cPPs and a CAVP certificate provides test evidence.

  Otherwise, the SFR is listed without a following /AA or /EE indicating the SFR applies to both cPPs and are identical.
- The SFR names have been modified. *(FDE_AA), (FDE_EE),* or *(FDE_AA) (FDE_EE)* ha been appended to the name to indicate which cPP was the source of the SFR.
- Where operations were completed in the cPP and relevant EPs/Modules/Packages/TDs, the formatting used in the source document has been retained except for text within brackets. If the text within brackets is completing an operation by the ST author, the conventions from the first three bullets apply. Otherwise, the text is presented in plain text.
- Additionally, all strikeout text formatting from the cPPs and relevant EPs/Modules/Packages/TDs is not retained in the ST.

5.1.3 Security Functional Requirements (SFRs)

The TOE functional requirements for this ST are taken directly from the cPPs which are derived from Common Criteria for Information Technology Security Evaluation Part 2, Version 3.1, Revision 5.

Table 6: Security Functional Requirements

Class	Component	Description
Cryptographic Support (FCS)	FCS_AFA_EXT.1	Authorization Factor Acquisition *(FDE_AA)*
	FCS_AFA_EXT.2	Timing of Authorization Factor Acquisition *(FDE_AA)*
	FCS_CKM.1(b)	Cryptographic Key Generation (Symmetric keys) *(FDE_EE)*
	FCS_CKM.1(c)	Cryptographic Key Generation (Data Encryption Key) *(FDE_EE)*
	FCS_CKM.4(a)/AA	Cryptographic Key Destruction (Power Management)

Class	Component	Description
		(FDE_AA)
	FCS_CKM.4(a)/EE	Cryptographic Key Destruction (Power Management) *(FDE_EE)*
	FCS_CKM.4(b)	Cryptographic Key Destruction (TOE-Controlled Hardware) *(FDE_EE)*
	FCS_CKM.4(d)	Cryptographic Key Destruction (Software TOE, 3rd Party Storage) *(FDE_AA)*
	FCS_CKM_EXT.4(a)	Cryptographic Key and Key Material Destruction (Destruction Timing) *(FDE_AA) (FDE_EE)*
	FCS_CKM_EXT.4(b)/AA	Cryptographic Key and Key Material Destruction (Power Management) *(FDE_AA)*
	FCS_CKM_EXT.4(b)/EE	Cryptographic Key and Key Material Destruction (Power Management) *(FDE_EE)*
	FCS_CKM_EXT.6	Cryptographic Key Destruction Types *(FDE_EE)*
	FCS_COP.1(a)/AA	Cryptographic Operation (Signature Verification) *(FDE_AA)*
	FCS_COP.1(a)/EE	Cryptographic Operation (Signature Verification) *(FDE_EE)*
	FCS_COP.1(b)/AA	Cryptographic Operation (Hash Algorithm) *(FDE_AA)*
	FCS_COP.1(b)/EE	Cryptographic Operation (Hash Algorithm) *(FDE_EE)*
	FCS_COP.1(c)	Cryptographic Operation (Keyed Hash Algorithm) *(FDE_EE)*
	FCS_COP.1(d)	Cryptographic Operation (Key Wrapping) *(FDE_EE)*
	FCS_COP.1(f)	Cryptographic Operation (AES Data Encryption/Decryption) *(FDE_EE)*
	FCS_KDF_EXT.1	Cryptographic Key Derivation *(FDE_EE)*
	FCS_KYC_EXT.1	Key Chaining (Initiator) *(FDE_AA)*
	FCS_KYC_EXT.2	Key Chaining (Recipient) *(FDE_EE)*
	FCS_RBG_EXT.1	Random Bit Generation *(FDE_EE)*
	FCS_SNI_EXT.1/AA	Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) *(FDE_AA)*
	FCS_SNI_EXT.1/EE	Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) *(FDE_EE)*
	FCS_VAL_EXT.1	Validation *(FDE_EE)*
User Data Protection (FDP)	FDP_DSK_EXT.1	Protection of Data on Disk *(FDE_EE)*
Security Management (FMT)	FMT_MOF.1	Management of Functions Behavior *(FDE_AA)*
	FMT_SMF.1/AA	Specification Management Functions *(FDE_AA)*

Class	Component	Description
	FMT_SMF.1/EE	Specification Management Functions *(FDE_EE)*
	FMT_SMR.1	Security Roles *(FDE_AA)*
Protection of the TSF (FPT)	FPT_FUA_EXT.1	Firmware Update Authentication *(FDE_EE)*
	FPT_KYP_EXT.1/AA	Protection of Key and Key Material *(FDE_AA)*
	FPT_KYP_EXT.1/EE	Protection of Key and Key Material *(FDE_EE)*
	FPT_PWR_EXT.1/AA	Power Savings State *(FDE_AA)*
	FPT_PWR_EXT.1/EE	Power Savings State *(FDE_EE)*
	FPT_PWR_EXT.2/AA	Timing of Power Savings States *(FDE_AA)*
	FPT_PWR_EXT.2/EE	Timing of Power Savings States *(FDE_EE)*
	FPT_TST_EXT.1/AA	TSF Testing *(FDE_AA)*
	FPT_TST_EXT.1/EE	TSF Testing *(FDE_EE)*
	FPT_TUD_EXT.1/AA	Trusted Update *(FDE_AA)*
	FPT_TUD_EXT.1/EE	Trusted Update *(FDE_EE)*

5.1.3.1 Cryptographic Support (FCS)

5.1.3.1.1 FCS_AFA_EXT.1 Authorization Factor Acquisition *(FDE_AA)*

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

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

].

Application Note: Addressed TD0759.
Application Note: [FDE_AA] Line 7&8, definition of a submask: "The TOE may accept any number of authorization factors, and these are categorized as 'submasks'".

5.1.3.1.2 FCS_AFA_EXT.2 Timing of Authorization Factor Acquisition *(FDE_AA)*

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

Refinement Rationale: The SFR was modified to reflect the SFR conventions followed by the ST as described in section 5.1.2.

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

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

5.1.3.1.4 FCS_CKM.1(c) Cryptographic Key Generation (Data Encryption Key) *(FDE_EE)*

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

- accept a DEK that is wrapped as specified in FCS_COP.1(d)

] and specified cryptographic key sizes [256 bits].

5.1.3.1.5 FCS_CKM.4(a)/AA Cryptographic Key Destruction (Power Management) *(FDE_AA)*

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

Refinement Rationale: The SFR was modified to reflect the SFR conventions followed by the ST as described in section 5.1.2.

5.1.3.1.6 FCS_CKM.4(a)/EE Cryptographic Key Destruction (Power Management) *(FDE_EE)*

FCS_CKM.4.1(a)/EE **Refinement:** The TSF shall [erase] **cryptographic keys and key material from volatile memory when transitioning to a Compliant power saving state as defined by FPT_PWR_EXT.1/EE** that meets the following: [a key destruction method specified in FCS_CKM_EXT.6].

Refinement Rationale: The SFR was modified to reflect the SFR conventions followed by the ST as described in section 5.1.2.

5.1.3.1.7 FCS_CKM.4(b) Cryptographic Key Destruction (TOE-Controlled Hardware) *(FDE_EE)*

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

- For volatile memory, the destruction shall be executed by a [
  - single overwrite consisting of [
    - zeroes;

    ]

  ]
- For non-volatile memory [
  - that employs a wear-leveling algorithm, the destruction shall be executed by a [
    - single overwrite consisting of zeroes

    ];

] that meets the following: [no standard].

Application Note: Volatile memory (RAM) is overwritten with zeroes except SATA registers, holding the DEKs. In addition to being overwritten with zeroes the SATA registers are overwritten with a new DEK upon invocation of the Administrator's "Change the DEK" command.

Application Note: The EE Component non-volatile memory includes the TSEM Flash and the RSM's EEPROM. Although EEPROM is not inherently a wear-leveling device, it has been designed to manage wear through techniques like wear leveling and therefore, considered a wear-leveling device.

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

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

- For volatile memory, the destruction shall be executed by a [
  - single overwrite consisting of [
    - zeroes,

  ]

]

] that meets the following: [no standard].

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

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

5.1.3.1.10 FCS_CKM_EXT.4(b)/AA Cryptographic Key and Key Material Destruction (Power Management) *(FDE_AA)*

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

Refinement Rationale: The SFR was modified to reflect the SFR conventions followed by the ST as described in section 5.1.2.

5.1.3.1.11 FCS_CKM_EXT.4(b)/EE Cryptographic Key and Key Material Destruction (Power Management) *(FDE_EE)*

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

Refinement Rationale: The SFR was modified to reflect the SFR conventions followed by the ST as described in section 5.1.2.

5.1.3.1.12 FCS_CKM_EXT.6 Cryptographic Key Destruction Types *(FDE_EE)*

FCS_CKM_EXT.6.1 The TSF shall use [FCS_CKM.4(b)] key destruction methods.

5.1.3.1.13 FCS_COP.1(a)/AA Cryptographic Operation (Signature Verification) *(FDE_AA)*

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

- RSA Digital Signature Algorithm with a key size (modulus) of [3072-bits]

] that meet the following: [

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

].

5.1.3.1.14 FCS_COP.1(a)/EE Cryptographic Operation (Signature Verification) *(FDE_EE)*

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

- Elliptic Curve Digital Signature Algorithm with a key size of 256 bits or greater

] that meet the following: [

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

].

5.1.3.1.15 FCS_COP.1(b)/AA Cryptographic Operation (Hash Algorithm) *(FDE_AA)*

FCS_COP.1.1(b)/AA **Refinement**: The TSF shall perform [cryptographic hashing services] in accordance with a specified cryptographic algorithm [SHA-384]–that meet the following: [ISO/IEC 10118-3:2004].

5.1.3.1.16 FCS_COP.1(b)/EE Cryptographic Operation (Hash Algorithm) *(FDE_EE)*

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

5.1.3.1.17 FCS_COP.1(c) Cryptographic Operation (Message Authentication) *(FDE_EE)*

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

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

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

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

FCS_COP.1.1(f) **Refinement:** The TSF shall perform [data encryption and decryption] in accordance with a specified cryptographic algorithm [AES used in [XTS] mode]

and cryptographic key sizes [256 bits] that meet the following: [AES as specified in ISO/IEC 18033-3, [XTS as specified in IEEE 1619]].

5.1.3.1.20 FCS_KDF_EXT.1 Cryptographic Key Derivation *(FDE_EE)*

FCS_KDF_EXT.1.1 The TSF shall accept [imported submask] to derive an intermediate key, as defined in [

- NIST SP 800-108 [KDF in Counter Mode]

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

Application Note: The first selection does not apply to this evaluation. The EE component does not derive (KBKDF) keys from a submask. The EE component derives keys from keys.

5.1.3.1.21 FCS_KYC_EXT.1 Key Chaining (Initiator) *(FDE_AA)*

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

- one, using a submask as the BEV

] while maintaining an effective strength of [256 bits] for symmetric keys and an effective strength of [not applicable] for asymmetric keys.

FCS_KYC_EXT.1.2 The TSF shall provide at least a [256 bit] BEV to [*the TSEM*] [

- without validation taking place

].

Application Note: this is the key chain from the KTD's authentication factor to the BEV.

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

FCS_KYC_EXT.2.1 The TSF shall accept a BEV of at least [256 bits] from [the AA].

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

- symmetric key generation as specified in FCS_CKM.1(b),
- derivation as specified in FCS_KDF_EXT.1,
- key wrapping as specified in FCS_COP.1(d)]

] while maintaining an effective strength of [256 bits] for symmetric keys and an effective strength of [not applicable] for asymmetric keys.

Application Note: this is the key chain from the BEV to the DEK.

5.1.3.1.23 FCS_RBG_EXT.1 Random Bit Generation *(FDE_EE)*

FCS_RBG_EXT.1.1 The TSF shall perform all deterministic random bit generation services in accordance with [[NIST SP 800-90A]] using [Hash_DRBG (any)].

FCS_RBG_EXT.1.2 The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from [

- [*1*] hardware-based noise source(s)

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

5.1.3.1.24 FCS_SNI_EXT.1/AA Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) *(FDE_AA)*

FCS_SNI_EXT.1.1/AA The TSF shall [use no salts].

FCS_SNI_EXT.1.2/AA The TSF shall use [no nonces].

FCS_SNI_EXT.1.3/AA The TSF shall [use no IVs].

Application Note: This SFR does not prescribe when salts, nonces, and IVs must be used, only that when they are used, they must be generated in a certain manner.
Application Note: Applied TD0760.

5.1.3.1.25 FCS_SNI_EXT.1/EE Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) *(FDE_EE)*

FCS_SNI_EXT.1.1/EE The TSF shall [use salts that are generated by a [DRBG as specified in FCS_RBG_EXT.1]].

FCS_SNI_EXT.1.2/EE The TSF shall use [no nonces].

FCS_SNI_EXT.1.3/EE The TSF shall create IVs in the following manner [

- XTS: No IV. Tweak values shall be non-negative integers, assigned consecutively, and starting at an arbitrary non-negative integer]

].

5.1.3.1.26 FCS_VAL_EXT.1 Validation *(FDE_EE)*

FCS_VAL_EXT.1.1 The TSF shall perform validation of the [BEV] using the following method(s): [

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

];

FCS_VAL_EXT.1.2 The TSF shall require the validation of the [BEV] prior to [allowing access to TSF data after exiting a Compliant power saving state].

FCS_VAL_EXT.1.3 The TSF shall [

- require power cycle/reset the TOE after [*three*] of consecutive failed validation attempts]

].

5.1.3.2 User Data Protection (FDP)

5.1.3.2.1 FDP_DSK_EXT.1 Protection of Data on Disk *(FDE_EE)*

FDP_DSK_EXT.1.1 The TSF shall perform Full Drive Encryption in accordance with FCS_COP.1(f), such that the drive contains no plaintext protected data.

FDP_DSK_EXT.1.2 The TSF shall encrypt all protected data without user intervention.

5.1.3.3 Security Management (FMT)

5.1.3.3.1 FMT_MOF.1 Management of Functions Behavior *(FDE_AA)*

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

5.1.3.3.2 FMT_SMF.1/AA Specification of Management Functions *(FDE_AA)*

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

a) forwarding requests to change the DEK to the EE,
b) forwarding requests to cryptographically erase the DEK to the EE,
c) allowing authorized users to change authorization values or set of authorization values used within the supported authorization method,
d) initiate TOE firmware/software updates,
e) [no other functions]

].

Application Note: Applied TD0767.

5.1.3.3.3 FMT_SMF.1/EE Specification of Management Functions *(FDE_EE)*

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

a) change the DEK, as specified in FCS_CKM.1, when re-provisioning or when commanded,
b) erase the DEK, as specified in FCS_CKM.4(a)**/EE**,
c) initiate TOE firmware/software updates,
d) [no other functions].

].

Refinement Rationale: The SFR was modified to reflect the SFR conventions followed by the ST as described in section 5.1.2.

Application Note: The un-iterated FCS_CKM.1 in this SFR is interpreted as FCS_CKM.1(b) Cryptographic Key Generation (Symmetric keys) (FDE_EE).

5.1.3.3.4 FMT_SMR.1 Security Roles *(FDE_AA)*

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

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

Application Note: This SFR refers to iECDL Administrators.

5.1.3.4 Protection of the TSF (FPT)

5.1.3.4.1 FPT_KYP_EXT.1/AA Protection of Key and Key Material *(FDE_AA)*

FPT_KYP_EXT.1.1/AA The TSF shall [

- Only store plaintext keys that meet anyone of the following criteria [

- the plaintext key will no longer provide access to the encrypted data after initial provisioning

]

].

Application Note: Applied TD0769.

5.1.3.4.2 FPT_KYP_EXT.1/EE Protection of Key and Key Material *(FDE_EE)*

FPT_KYP_EXT.1.1/EE The TSF shall [

- only store keys in non-volatile memory when wrapped, as specified in FCS_COP.1(d), or encrypted, as specified in FCS_COP.1(g) or FCS_COP.1(e)

].

Application Note: Addressed TD0769.

Application Note: FCS_COP.1(d) Cryptographic Operation (Key Wrapping) (FDE_EE) is claimed in this ST (FCS_COP.1(g) Cryptographic Operation (Key Encryption) (FDE_EE) and FCS_COP.1(e) Cryptographic Operation (Key Transport) (FDE_EE) are not claimed. Therefore, this SFR applies to FCS_COP(d) Cryptographic Operation (Key Wrapping) (FDE_EE)).

5.1.3.4.3 FPT_PWR_EXT.1/AA Power Saving States *(FDE_AA)*

FPT_PWR_EXT.1.1/AA The TSF shall define the following Compliant power saving states: [G2(S5), G3].

5.1.3.4.4 FPT_PWR_EXT.1/EE Power Saving States *(FDE_EE)*

FPT_PWR_EXT.1.1/EE The TSF shall define the following Compliant power saving states: [G2(S5), G3].

Application Note: Updated as per TD0464.

5.1.3.4.5 FPT_PWR_EXT.2/AA Timing of Power Saving States *(FDE_AA)*

FPT_PWR_EXT.2.1/AA For each Compliant power saving state defined in FPT_PWR_EXT.1.1**/AA**, the TSF shall enter the Compliant power saving state when the following conditions occur: user-initiated request, [[*loss of power*]].

Refinement Rationale: The SFR was modified to reflect the SFR conventions followed by the ST as described in section 5.1.2.

5.1.3.4.6 FPT_PWR_EXT.2/EE Timing of Power Saving States *(FDE_EE)*

FPT_PWR_EXT.2.1/EE For each Compliant power saving state defined in FPT_PWR_EXT.1.1**/EE**, the TSF shall enter the Compliant power saving state when the following conditions occur: user-initiated request, [[*loss of power*]].

Refinement Rationale: The SFR was modified to reflect the SFR conventions followed by the ST as described in section 5.1.2.

5.1.3.4.7 FPT_TST_EXT.1/AA TSF Testing *(FDE_AA)*

FPT_TST_EXT.1.1/AA The TSF shall run a suite of the following self-tests [during initial start-up (on power on)] to demonstrate the correct operation of the TSF: [

- *FIPS Power-On Self Tests*

].

5.1.3.4.8 FPT_TST_EXT.1/EE TSF Testing *(FDE_EE)*

FPT_TST_EXT.1.1/EE The TSF shall run a suite of the following self-tests [during initial start-up (on power on)] to demonstrate the correct operation of the TSF: [

- *ECDSA signature verification CAST (includes SHA-384),*
- *TOE firmware digital signature verification*
- *KBKDF key derivation CAST (includes HMAC and SHA-384),*
- *Key wrap and unwrap CAST (includes forward AES CAST),*
- *TSEM Cryptographic Library initialization automatic DRBG self-test,*
- *Lycan FPGA self-tests, which include*
  - *AES-XTS encryption CAST (including embedded forward and inverse AES),*
  - *AES-XTS decryption CAST (including embedded forward and inverse AES),*

].

5.1.3.4.9 FPT_TUD_EXT.1/AA Trusted Update *(FDE_AA)*

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

FPT_TUD_EXT.1.2/AA **Refinement**: The TSF shall provide [authorized users] the ability to initiate updates to TOE [software].

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

Refinement Rationale: The SFR was modified to reflect the SFR conventions followed by the ST as described in section 5.1.2.

5.1.3.4.10 FPT_TUD_EXT.1/EE Trusted Update *(FDE_EE)*

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

FPT_TUD_EXT.1.2/EE **Refinement:** The TSF shall provide [authorized users] the ability to initiate updates to TOE [firmware]**.**

FPT_TUD_EXT.1.3/EE **Refinement:** The TSF shall verify updates to the TOE [firmware] using a [authenticated firmware updated mechanism as describe in FPT_FUA_EXT.1] by the manufacturer prior to installing those updates.

Refinement Rationale: The SFR was modified to reflect the SFR conventions followed by the ST as described in section 5.1.2.

5.1.3.4.11 FPT_FUA_EXT.1 Firmware Update Authentication *(FDE_EE)*

FPT_FUA_EXT.1.1 The TSF shall authenticate the source of the firmware update using the digital signature algorithm specified in FCS_COP.1(a)**/EE** using the RTU that contains [the public key].

FPT_FUA_EXT.1.2 The TSF shall only allow installation of update if the digital signature has been

	successfully verified as specified in FCS_COP.1(a)**/EE**.
FPT_FUA_EXT.1.3	The TSF shall only allow modification of the existing firmware after the successful validation of the digital signature, using a mechanism described in FPT_TUD_EXT.1.2**/EE**.
FPT_FUA_EXT.1.4	The TSF shall return an error code if any part of the firmware update process fails.

Refinement Rationale: The SFR was modified to reflect the SFR conventions followed by the ST as described in section 5.1.2.

5.1.4 TOE SFR Dependencies Rationale for SFRs

The cPPs contain all the requirements claimed in this ST. As such, dependencies are not applicable since the cPPs have been approved.

5.2 Security Assurance Requirements

5.2.1 Extended Assurance Requirements

There are no extended assurance requirements defined in the FDE_AA or the FDE_EE cPPs and therefore, none are included in this ST.

5.2.2 Security Assurance Requirements (SARs)

The TOE assurance requirements for this ST are taken directly from the PP and any relevant EPs/Modules/Packages, which is/are derived from Common Criteria Version 3.1, Revision 5. The assurance requirements are summarized and the operations on ASE_TSS.1 implemented in the Table 7.

The TOE assurance requirements for this ST are taken directly from the cPPs which are derived from Common Criteria for Information Technology Security Evaluation Part 3, Version 3.1, Revision 5.

Table 7: Security Assurance Requirements

Assurance Class	Assurance Components	Component Description
Security Target	ASE_CCL.1	Conformance Claims
	ASE_ECD.1	Extended Components Definition
	ASE_INT.1	ST Introduction
	ASE_OBJ.1	Security Objectives for the Operational Environment
	ASE_REQ.1	Stated Security Requirements
	ASE_SPD.1	Security Problem Definition
	ASE_TSS.1	TOE Summary Specification (Refined) ASE_TSS.1.1C Refinement: The TOE summary specification shall describe how the TOE meets each SFR, including a proprietary Key Management Description (Appendix E), and [no other cPP specified proprietary documentation].
Development	ADV_FSP.1	Basic Functionality Specification
Guidance Documents	AGD_OPE.1	Operational User Guidance

Assurance Class	Assurance Components	Component Description
	AGD_PRE.1	Preparative Procedures
Life Cycle Support	ALC_CMC.1	Labelling of the TOE
	ALC_CMS.1	TOE CM coverage
Tests	ATE_IND.1	Independent Testing – Sample
Vulnerability Assessment	AVA_VAN.1	Vulnerability Survey

5.2.3 Assurance Measures

The TOE satisfied the identified assurance requirements. This section identifies the Assurance Measures applied by Ampex Data Systems Corporation to satisfy the assurance requirements. Table 8 lists the details.

Table 8: TOE Security Assurance Measures

SAR Component	How the SAR will be met
ASE_TSS.1	**Refinement**: The TOE summary describes how the TOE meets each SFR. It also includes a reference to the proprietary Key Management Description (Appendix E) and [Entropy Essay].
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 may 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_OPE.1	The operational user guidance does not have to be contained in a single document. Guidance to users, administrators, and integrators 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.
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.
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
ALC_CMS.1	Given the scope of the TOE and its associated evaluation evidence requirements, the

SAR Component	How the SAR will be met
	evaluator performs the CEM work units associated with ALC_CMS.1.
ATE_IND.1	Ampex will provide the TOE for testing.
AVA_VAN.1	Ampex will provide the TOE for testing. Ampex will provide a document identifying the list of software and hardware components.

5.2.4 Rationale for Security Assurance Requirements

The functional specification describes the external interfaces of the TOE, such as the means for a user to invoke a service and the corresponding response of those services. The description includes the interface(s) that enforces a security functional requirement, the interface(s) that supports the enforcement of a security functional requirement, and the interface(s) that does not enforce any security functional requirements. The interfaces are described in terms of their purpose (general goal of the interface), method of use (how the interface is to be used), parameters (explicit inputs to and outputs from an interface that control the behavior of that interface), parameter descriptions (tells what the parameter is in some meaningful way), and error messages (identifies the condition that generated it, what the message is, and the meaning of any error codes). The development evidence also contains a tracing of the interfaces to the SFRs described in this ST.

6 TOE Summary

This chapter identifies and describes how the Security Functional Requirements (SFRs) identified above are met by the TOE.

6.1 Cryptographic Support (FCS)

6.1.1 FCS_AFA_EXT.1 Authorization Factor Acquisition (FDE_AA)

The TSF supports one authorization factor, a KEK (Key Encryption Key) wrapped DEK (Data Encryption Key) (KEK[DEK]). This authentication factor is stored on the KTD (Key Transfer Device) or a cRSM (compact Removable Storage Media). Both devices are USB drives and the iECDL supports a USB port for each device. They are stored in non-volatile memory on both USB devices and indexed by the serial number of the RSMs the iECDL's customer owns.

The KEK is a 256 bit derived key material (KBKDF). The DEK is two 256 bit AES-XTS symmetric keys concatenated together and processed as a single 512 bit entity. The DEK is wrapped using the KEK, in accordance with the key wrap defined in Sect. 6.2 of NIST SP 800-38F, and block ciphers as specified in ISO/IEC 18033-3. The key wrap uses 256-bit AES KW using a KBKDF with HMAC-SHA-384 as pseudorandom function. The wrapped key is 256 bits with the security strength of 256.

The TOE's auth factors are generated by the TOE. The system does not support external generated auth factors.

The iECDL is shipped with a KTD populated with auth factors for each of the iECDL the customer ordered and each iECDL is shipped with the RSM inserted. Ampex engineers use the soon to be delivered iECDL to populate the KTD, with the installed RSM, by issuing the "Initialize RSM" GUI command. This command supports the "Modify Authentication Factor" mandatory Administrative command by creating DEKs for the specific installed RSM and storing the DEKs on the KTD. To populate a cRSM with an auth factor, the Administrator issues the "Sync" iECDL GUI command. This command will duplicate a the KTD.

Additionally, the TOE supports the mandatory "Modify the Authentication Factor" management command. To invoke this, the Administrator selects the "Initialize RSM" or "Reinitialize RSM" iECDL GUI command. Both commands will instruct the EE component to reprovision the TSEM, create new DEKs, wrap the new DEK with a new KEK, and pass the new auth factor to the AA to be stored on the KTD.

The auth factor is read by the Host System (AA) and passed to the TSEM (EE) for processing and use. The auth factor is the BEV passed to the EE component of the TOE and is 256 bits. The auth factor is also the submask[3], referred to in the SFRs.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.1.2 FCS_AFA_EXT.2 Timing of Authorization Factor Acquisition (FDE_AA)

The TOE's authentication factor is a KEK (Key Encrypting Key) wrapped DEK (Data Encrypting Key) stored on the KTD or cRSM (KEK[DEK]). The KEK is a 256 bit AES key. The DEK is two 256 bit AES-XTS symmetric keys concatenated together and processed as a single 512 bit entity. The DEK is wrapped by the EE

[3] *[FDE_AA] Line 7&8, definition of a submask: "The TOE may accept any number of authorization factors, and these are categorized as 'submasks'".*

component of TOE that implements key wrapping/unwrapping functions using 256-bit AES in accordance with the key wrap defined in Sect. 6.2 of NIST SP 800-38F and ISO/IEC 18033-3. The auth factors are stored on the KTD or cRSM, indexed by the serial number of the RSMs.

The auth factor is read from the KTD or cRSM when the system boots from a Compliant power saving state and KTD or cRSM is inserted. If a KTD or cRSM are not inserted, the iECDL continues operating but Administrators and Users do not have access to the protected data stored on the RSM. If a KTD or cRSM is inserted and an RSM is inserted, the AA component will search the USB devices for the auth factor associated with the serial number of the RSM. If found, the auth factor will be passed to the EE component for validation. If there is not an auth factor for the serial number, the iECDL continues operating but Administrators and Users do not have access to the protected data stored on the RSM.

If the EE validates the auth factor successfully, Administrators and Users have access to the protected data stored on the RSM. If the EE validation fails, access is denied until the correct auth factor is installed on a KTD/cRSM.

Validation always occurs when an auth factor is read from a KTD or cRSM and all auth factors are validated.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

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

The EE component of the TOE generates two symmetric keys. Both are used in the EE side key chain.

- The system's TMK (TSEM Master Key) a 256 bit symmetric key created when the iECDL is provisioned and
- the two 256 bit AES-XTS DEKs (Data Encryption Keys) created when the Administrator issues the "Change Authorization Factor" command.

The TMK is created when the iECDL is provisioned. Provisioning is the process of creating the keys and key material that are in the key chain. The key chain enables the system to recreate the KEK used to protect the DEK stored on the KTD. Recreating the KEK and comparing it to the KEK of the auth factor is the validation process. The TMK is stored encrypted (wrapped) in TSEM non-volatile memory (Flash).

The DEKs are also created when the iECDL is initialize or reinitialized (the Create_Keys TSEM function). They are used to create the authentication factor. During the Create_Keys() TSEM command they wrapped (protected) with the KEK derived during the initialize or reinitialized process, passed to the Host System, who stores the new auth factor protected (KEK wrapped DEK) on the KTD. During the Put_Keys() TSEM command they read from the KTD by the Host System, passed to the TSEM, and unwrapped with the KEK derived from the TSEM. If the unwrap returned a DEK, validation was successful. If the unwrap returned an error, validation failed. Once the initialization or reinitialization process ends, the TSEM erases (overwrites with zeroes) all keys and keying material used and stored in volatile memory.

The 256-bit key size of the TMK and the 256 bit size and the AES-XTS mode of the DEK are the default values and cannot be modified by an authorized administrator.

The keys, their origin, and their use is described in Section 6.6 Cryptographic Key Destruction in this document.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.1.4 FCS_CKM.1(c) Cryptographic Key Generation (Data Encryption Key) (FDE_EE)

The TOE creates its own DEKs (Data Encryption Key), utilizing the TSEM Crypto Library included in the TOE's physical boundary. The TOE does not rely on the Operational Environment to create the DEKs.

The iECDL's authorization factor is the protected DEK of the installed RSM. DEKs are stored wrapped, on either a KTD or a cRSM. The iECDL is shipped with a KTD populated with auth factors for each of the RSMs the customer ordered. Ampex engineers use the soon to be delivered iECDL to populate the shipped KTD. The system is shipped with the RSM or iRSM inserted, thereby offering immediate data protection without requiring any Administrator action.

Because the KTD is populated upon customer delivery, the iECDLs do not need to create DEKs except if the customer orders a new RSM and in support of the cPP mandatory administrative command, "Change the DEK" (FMT_SMF.1/EE Specification of Management Functions *(FDE_EE))*. When this command is selected ("Initialize/Reinitialize RSM" GUI Command), the EE component of the TOE will generate a new DEK and send it to the AA component where it is stored, wrapped, on the inserted KTD.

In the operational state, DEKs are held in the SATA Controller's registers (volatile memory) and used for encrypting and decrypting the user data stored on the attached drive (RSM). The DEKs are stored in plaintext however, because they are stored in the SATA Controller registers, they are considered protected. The SATA registers are protected from unauthorized access by Register Access Protection (RAP) and privilege levels. These mechanisms ensure that only authorized processes can access the SATA controller registers, preventing unintended writes or reads.

The EE will generate the DEK's two 256-bit AES-XTS symmetric keys by calling the TSEM Crypto Library `EVP_EncryptInit_ext()` function. This function is called with the parameters specifying AES, 256, and XTS. The routine returns the two concatenated AES keys for XTS and the tweak value.

The `EVP_EncryptInit_ext()` function does not directly involve a random bit generator (RBG). Instead, the function initializes a cipher context for encryption, which is a cryptographic operation that does not inherently require randomness. The `EVP_EncryptInit_ext()` function sets up the cipher context for encryption, but it does not include any random number generation as part of its functionality.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.1.5 FCS_CKM.4(a)/AA Cryptographic Key Destruction (Power Management) (FDE_AA)

Keys and Key Material (KM) are erased from the Host System's volatile memory by the TOE's AA component (ACCE) when no longer used. The AA component uses RHEL's VMM (Virtual Memory Manager) to store, retrieve, and destroy values in RAM. Destroy involves overwriting the key or KM with zeroes. Memory locations in the application (ACCE) are tracked through a system of virtual addresses, which are unique identifiers for specific memory or storage locations.

Refer to in Section 6.6 Cryptographic Key Destruction of this document for a full description of key destruction.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.1.6 FCS_CKM.4(a)/EE Cryptographic Key Destruction (Power Management) (FDE_EE)

Keys and key material are erased from the TSEM's volatile memory by the TOE's EE component (TSEM Firmware) when no longer used. The TOE boundary of the TSEM card is the hardware and software of the card, so there is no Operational Environment to rely on to erase volatile memory. Memory locations in the application (TSEM) are tracked through a system of virtual addresses, which are unique identities for specific memory or storage locations. Keys and key material are overwritten with zeroes in order to destroy them.

Refer to in Section 6.6 of this document for a full description of key destruction.

Refer to the developer's proprietary annex *Key Management Description* associated with this ST for a detail description of the TOE implementing the above commands. The annex is made available at the discretion of the developer.

6.1.7 FCS_CKM.4(b) Cryptographic Key Destruction (TOE-Controlled Hardware) *(FDE_EE)*

The TSEMs volatile memory includes FPGA RAM, Secure MCU RAM, and the SATA registers which hold the DEKs. All keys and secure elements used by the TSEM component of the TOE (EE component) and stored in volatile memory are erased immediately after use.

The keys managed in TSEM's volatile memory are received by the AA component, derived from another key or unwrapped by another key. The keys are the DEK (Data Encrypting Key), KEK (Key Encrypting Key), PIK (Platform Identity Key), TIK (TSEM Identity Key), TMK (TSEM Master Key). Refer to Table 14: The iECDL Keys and Key Material Lifecycle in Volatile Memory, column three, for a description of how the keys are introduced into volatile memory and column six for how the key is destroyed.

Table 14 and Table 15 lists each type of key that is stored and identifies the memory type where key material is stored in volatile and non-volatile memory respectively.

Volatile memory is overwritten with zeroes. The exception is SATA registers are overwritten with a new DEK upon invocation of the Administrator's "Change the DEK" command. The SATA registers are overwritten with zeroes upon invocation of the "Erase the DEK" command, invocation of the "Shutdown" iECDL GUI command, and loss of power by any means.

The TSEMs non-volatile memory includes the TSEM's Flash, a wear-leveling device. The TSEM's Flash holds the TIK wrapped TMK (TIK[TMK]) and the Salt key material. Both are overwritten with zeroes upon invocation of the "Erase the DEK" command.

The RSM includes an EEPROM as non-volatile memory. This EEPROM stores an alphanumeric string which is the RSM's serial number. It is used to search for authentication factors (BEVs) on the KTDs. This alphanumeric string is created during production and remains in the EEPROM the lifetime of the TOE. Additionally, the EEPROM contains a QEE, key material stored or overwritten when the iECDLs are Provisioned. The QEE is overwritten with zeroes when an Administrator issues a "Erase the DEK" command.

RAM is organized in cells, verses blocks, which are the fundamental building blocks of RAM memory. Each memory cell stores one bit of binary information, and RAM consists of multiple memory cells arranged in a grid format, allowing for efficient data access and storage.

The TSEM's memory controller utilizes the Memory Allocation method that provides efficient distributed memory resources across memory modules to optimize performance.

There are no circumstances that will prevent the keys or keying material from being erased.

Refer to Section 6.6 of this document for a full description of key destruction.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

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

The authentication factors are introduced into the Host System's volatile memory (RAM) when:

- the AA receives a "Put Keys" TSEM command from the EE component of the TOE, and
- the AA receives a "Create_Keys" TSEM command from the EE component of the TOE.

The ACCE communicates with the TSEM firmware via a USB connection. The ACCE includes a library, the TSEM Library, that contains the APIs to communicate with the EE component. Likewise, the TSEM's Host System Library provides the APIs to communicate with the AA component. Refer to Figure 2: iECDL TOE Components for detail.

The auth factor is a KEK (Key Encryption Key) wrapped DEK (Data Encryption Key) (KEK[DEK]). The KEK is a 256 bit derived key (KBKDF). The DEK is two 256 bit AES-XTS symmetric keys concatenated together and processed as a single 512 bit entity. The DEK is wrapped using the KEK, in accordance with the key wrap defined in Sect. 6.2 of NIST SP 800-38F, and block ciphers as specified in ISO/IEC 18033-3. The key wrap uses 256-bit AES KW using a KBKDF with HMAC-SHA-384 as pseudorandom function. The wrapped key is 256 bits with the security strength of 256.

The AA will receive a "Put_Keys" command from the EE when the system is Initialize or Reinistialize command. This command load the DEKs, held in the auth factor, into the SATA Controller registers. Upon receipt of the command, the AA component reads the serial number of the RSM, a 16 bit character string that is stored in the EEPROM of the RSM, from the RSM's EEPROM and searches for the serial number entry on the KTD. If found, the AA component reads the auth factor associated with the serial number and passes the auth factor to the EE component. Once sent, the auth factor and RSM serial number are overwritten with zeroes.

The AA will receive a "Create_Keys" command from the EE when the EE is processing the "Change the Authentication Factor" command. The command includes a new auth factor. Upon receipt, the AA reads the serial number of the RSM from the RSM's EEPROM, searches for the serial number entry on the KTD, and overwrites the auth factor stored at that index with the new auth factor. Once stored, the new auth factor and RSM serial number are overwritten with zeroes.

Non-volatile memory of the Host System includes a key stored in non-volatile memory. This key, the PIK (Platform Identity Key), is a 256-bit AES key created at production time and stored in the Host System's Trusted Platform Module (TPM) by the manufacturer (Infineon). The PIK is passed to the EE component and used to derive the TIK (TSEM Identity Key). The AA component will pass the PIK to the EE upon receipt of a "Provision", "Create Keys", and "Load Keys" command. Once passed, the AA component erases the

key from volatile memory because it is no longer needed. The PIK held in non-volatile memory is never overwritten, modified, or cleared by the TSF as it's an identification key, used to identify the TPM, generated by the manufacturer, and exists for the lifetime of the TPM.

The iECDL's RHEL OS includes the tpm-tools package. This package enables application programs to read values from the TPM. The AA component uses RHEL's VMM (Virtual Memory Manager) to store, retrieve, and destroy values in RAM. The AA component uses the `eeprom()` call which is included in the RHEL's EEPROM library to read the RSM's EEPROM.

All keys used by the AA component of the TOE and stored in volatile memory are erased immediately after use. There are no circumstances or configuration that will prevent the keys from being erased.

Refer to Section 6.6 of this document for a description of key destruction.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.1.9 FCS_CKM_EXT.4(a) Cryptographic Key and Key Material Destruction (Destruction Timing) (FDE_AA) (FDE_EE)

The TOE destroys all keys and key material (KM) when no longer needed.

The AA component of the TOE considers a key or KM held in the Host System's non-volatile memory no longer needed when the ACCE is finished using it for processing i.e. used it as an input, sent to the EE, or stored in non-volatile memory.

The EE component of the TOE considers a key or KM, held in the TSEM's RAM, no longer needed when the TSEM FW is finished using it for processing. The use by the EE component is either to unwrap a key (decrypt), wrap a key (encrypt), derive a key (KBKDF), send a value to the AA, and save a value in non-volatile or volatile memory (DEKs in SATA Controllers).

Refer to Section 6.6 of this document for a description of key destruction.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.1.10 FCS_CKM_EXT.4(b)/AA Cryptographic Key and Key Material Destruction (Power Management) (FDE_AA) (FDE_EE)

The volatile memory of the Host System is the Intel® Atom® C3558 RAM (Random Access Memory). The volatile memory of the TSEM is the Secure MCU RAM, FPGA RAM, and the SATA registers.

With the exception of the SATA registers, the keys and key material stored in volatile memory are destroyed (overwritten with zeroes) after use by the either the AA component or the EE component. When the iECDL enters a Compliant power-saving state, all keys in memory have been cleared. The SATA registers are however cleared when entering a Compliant power saving state via the "shutdown" GUI selection or loss of power.

When a computer is shut down, the contents of RAM are cleared, and any sensitive data, such as encryption keys and key material, are no longer stored in the memory. This is because RAM is volatile memory type, meaning it requires a constant power supply to retain data. Once power is turned off, the RAM loses its contents and is reset, ready to be filed with new data when the computer restarts.

Therefore, keys and key material are not zeroed in RAM when shutdown, but they are not stored in RAM either.

Keys and key material remain in non-volatile memory when the system enters into a Compliant power saving state.

A summary of the cryptographic keys and their destruction is given in Section 6.6 in this document.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.1.11 FCS_CKM_EXT.6 Cryptographic Key Destruction Types (FDE_EE)

The TOE maintains two concatenated key chains, one for the AA component (Host System card) and one for the EE component (TSEM card). The AA's key chain begins at the auth factor and ends at the BEV (Border Encryption Key) that is passed to the EE. The EE's key chain is the BEV to the DEK (Data Encryption Key).

The TOE destroys keys and key material by overwriting them with zeroes. Both components of the TOE destroy all keys and keying material unwrapped, derived, read from non-volatile memory, or received after use.

The key chains and their processing are considered Ampex proprietary information. Therefore, refer to the *Key Management Description TuffServ® Intelligent E-2 Common Data Loader (iECDL) v2.74.1* associated with this ST for details of navigating the key chains. The annex is made available at the discretion of the developer.

6.1.12 FCS_COP.1(a)/AA Cryptographic Operation (Signature Verification) (FDE_AA)

The AA TSF's only use of signature verification is verifying the new software build's digital signature of the ACCE application code that runs on the Host System.

The TOE uses RPM (Red Hat Package Manager) to update the ACCE software. The update file is signed with an Ampex signing key which is an RSA 3072-bit key. An RPM file includes a .rpm extension. The format of the file is subject to change but always includes four distinct sections: the lead, the signature, the header, and the archive.

When a package is installed, the signature on the package is validated by the RPM tool, included in the Operational Environment (RHEL OS). The RPM includes digests of the files within the RPM. These digests are stored in a database on the system during package installation. During TOE upgrade, the contents of each file are verified against the stored digests.

iECDL Administrators will be notified by email if a new version of the ACCE has been released. The email will also contain the address of a secure site where the new version can be downloaded. The new version can be copied to the iECDL by a remote user using SCP (Secure Copy Protocol). Details of updating the AA software, ACCE, are provided in the CC User Guidance (AGD_CC).

The RPM digital signature verification primarily checks the critical portions of the package. The signature is associated with specific areas of the RPM file, and while the entire file can be verified, the RPM signature verification is limited to these critical sections.

In addition to digital signature verification, RPM also performs the following verifications:

- File attribute checks: Verifies at least nine attributes for each file, including:

  - File size
  - MD5 checksum
  - Permissions (mode)
  - Type
  - Owner and group
  - Device major/minor numbers
  - Symbolic link string
  - Modification time
- Dependency checks: Confirms that any other packages a package depends on are installed.

The cryptographic library used to verify the digital signature is AMPEX OpenSSL Cryptographic Module.

6.1.13 FCS_COP.1(a)/EE Cryptographic Operation (Signature Verification) (FDE_EE)

The TOE firmware is digitally signed by the Ampex during the production of the TOE. The TOE implements ECDSA P-384 digital signature verification on the SHA-384 message digest of the TOE firmware. The digital signature public key is stored on the non-volatile memory of the TSEM (Flash). The private key used for the digital signature computation is a production key only known to the Ampex production environment.

When the EE component of the TOE boots up, the boot sequence implements a suite of self-tests to ensure correct operation of the TOE. One of the self-tests is to verify the digital signature of the TOE firmware. The TOE implements signature verification using the TSEM Cryptographic Library, which is inside the TOE boundary. If the digital signature verification fails, the TOE enters a non-operational error state.

6.1.14 FCS_COP.1(b)/AA Cryptographic Operation (Hash Algorithm) (FDE_AA)

The Host System component of the TOE (AA) implements SHA-384 hashing for use in association with digital signature verification of an ACCE update. The cryptographic library is AMPEX OpenSSL Cryptographic Module.

6.1.15 FCS_COP.1(b)/EE Cryptographic Operation (Hash Algorithm) (FDE_EE)

The EE component of the TOE supports SHA-384 hashing in accordance to ISO/IEC 10118-3:2004, *Information technology – Security techniques – Hash-functions – Part 3: Dedicated hash-functions.*

The EE component of the TOE uses SHA-384 hashing for the following functions:

- Key derivation of the TIK (TSEM Identity Key) (FCS_KDF_EXT.1 Cryptographic Key Derivation *(FDE_EE)*).
- Key derivation of the KEK (Key Encryption Key) (FCS_KDF_EXT.1 Cryptographic Key Derivation *(FDE_EE)*).
- Key wrapping the TIK (TSEM Identity Key) (FCS_COP.1(d) Cryptographic Operation (Key Wrapping) *(FDE_EE)*).
- Key wrapping the DEK (Data Encryption Key). (FCS_COP.1(d)) Cryptographic Operation (Key Wrapping) *(FDE_EE)*).
- Digital signature verification of the firmware (FCS_COP.1(a)/EE Cryptographic Operation (Signature Verification) *(FDE_EE)*). *(FDE_EE)*, FPT_TUD_EXT.1/EE Trusted Update *(FDE_EE)*).

Note: Key unwrapping does not require a hash function, only wrapping.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.1.16 FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm) (FDE_EE)

The EE component of the TOE implements HMAC function that uses the SHA hash algorithm with a 256-bit key as specified in ISO/IEC 9797-2:2011, Section 7 “MAC Algorithm 2”. HMAC-SHA-384 and produces a 384-bit length hash value as output and uses a 576-bit block size. HMAC is used as input to calls to the EE component’s Key Based Key Derivation Function (KBKDF) used to navigate the key chain.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.1.17 FCS_COP.1(d) Cryptographic Operation (Key Wrapping) (FDE_EE)

The EE component of TOE implements key wrapping/unwrapping functions using 256-bit AES in accordance with the key wrap defined in Sect. 6.2 of NIST SP 800-38F, *Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping* and block ciphers as specified in ISO/IEC 18033-3, *Information technology – Security techniques – Encryption algorithms – Block ciphers.*

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.1.18 FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption) (FDE_EE)

The EE component of the TOE encrypts and decrypts user data stored on the RSM drives with a 256-bit AES in XTS mode of operation. AES-XTS uses two independent 256-bit symmetric keys. The two keys are stored as a single encrypted 512-bit entity in the SATA Controller Registers. The encryption and the corresponding decryption is performed by the FPGA components, specifically the SATA Controller which resides on the TSEM connected to the drive via a SATA bus.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.1.19 FCS_KDF_EXT.1 Cryptographic Key Derivation (FDE_EE)

The TOE implements Key-based Key Derivation (KBKD) as defined in NIST Special Publication 800-108, *Recommendation for Key Derivation Using Pseudorandom Functions*. The TOE does not implement Password-Based Key Derivation as defined in NIST Special Publication 800-132, *Recommendation for Password-Based Derivation*.

The EE component derives two keys: the TIK (TSEM Identity Key) and the KEK (Key Encryption Key), both using a KBKDF with HMAC-SHA-384 in Counter Mode, 10000 iterations, and a 256 bit Salt.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.1.20 FCS_KYC_EXT.1 Key Chaining (Initiator) (FDE_AA)

This SFR addresses the key chain managed by the Host System. Specifically, the chain of keys and key material from the authentication factor to the BEV (Border Encryption Value). The AA component of the TOE uses the authentication factor obtained from a Key Transfer Device (KTD) or cRSM (compact Removable Storage Media) as the BEV which results in the AA component maintaining a key chain of one. The BEV is a KEK (Key Encryption Key) wrapped DEK (Data Encryption Key) (KEK[DEK]) and is 256 bits in length.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex Key Management Description associated with this ST. The annex is made available at the discretion of the developer.

6.1.21 FCS_KYC_EXT.2 Key Chaining (Recipient) (FDE_EE)

There are no TSS evaluation activities for this SFR.

6.1.22 FCS_RBG_EXT.1 Random Bit Generation (FDE_EE)

The DRBG of the EE is implemented in firmware using the TSEM Cryptographic Library included in the TOE boundary. The generation uses one hardware entropy source and no software entropy sources. The hardware entropy source is the hardware-based RBG of the SoC (System on Chip) that seeds and reseeds with at least 256-bits of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011 Table C.1 “Security Strength Table for Hash Functions”, of the keys and hashes that it will generate.

The TOE uses the RBG function for generation of the following keys and key material:

- Create the Quick Erase Element (QEE), a 256 bit random number used as input to the derivation function (KBKDF) that derives the KEK (Key Encryption Key).
- Create the TSEM Master Key (TMK), a 256 bit symmetric key used in the key chain to derive a KEK (Key Encryption Key).
- Create the Salt, 128 bit random data used as input to the derivation function (KBKDF) that derives the TIK (TSEM Identity Key).
- Derive the KEK, a 256 bit key that is derived from the TSEM Master Key (TMK).

Note: the function that creates the DEKs, two 256-bit AES-XTS symmetric keys used for encryption/decryption, does not directly involve a random bit generator (RBG) (refer to Section 6.1.4 FCS_CKM.1(c) Cryptographic Key Generation (Data Encryption Key) *(FDE_EE)*).

A full description of the entropy source is described in a proprietary ancillary *Entropy Assessment Report* made available at the discretion of the developer.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.1.23 FCS_SNI_EXT.1/AA Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation*)* (FDE_AA)

The AA component of the TOE does not generate or require salt values, nonces, IVs (Initialization Vectors), or tweaks.

Application Note: This mandatory SFR does not prescribe when salts, nonces, IVs, and tweaks must be used, only when they are used, they must be generated in a certain manner. Salts, nonces, IVs, and tweaks are not used in the AA component.

6.1.24 FCS_SNI_EXT.1/EE Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) (FDE_EE)

The EE component of the TOE does not generate nonces, and uses no IVs.

The EE component of the TOE requires a Salt value, used as the context parameter to derive (Key-Based Key Derivation Function (KBKDF)) the TIK (TSEM Identity Key). The EE Salt is 128 bit random data, generated by a call to the `rand()` function. The `rand()` function generates cryptographically secure random bytes[4]. The TSEM Crypto Library's `rand()` function uses the TOE's random number generator. It utilizes a pseudo-random number generator (CSPRING) based on the deterministic random bit generator (DRBG) model.

The EE Component uses Tweak values for AES-XTS encryption and decryption of the data on the RSMs. The tweak value is a 128-bit value used to represent the logical position of the data being encrypted and decrypted. The tweak value is generated randomly and used to XOR the plaintext before the encryption process begins. The TSEM's tweaks are non-negative integers, assigned consecutively, and starting at an arbitrary non-negative integer. The EE tweak is generated by a call to the TSEM Crypto Library `EVP_EncryptInit_ext()` function. This function is called with the parameters specifying AES, 256, and XTS. The routine returns the two concatenated AES keys for XTS and the tweak value.

6.1.25 FCS_VAL_EXT.1 Validation (FDE_EE)

The TOE supports one type of authentication factor (auth factor): a KEK (Key Encryption Key) wrapped DEK (Data Encryption Key) (KEK[DEK]). Multiple submasks (authentication factors) are not used. The EE component of the TOE does not implement submask combining (FCS_SMC_EXT.1 Submask Combining *(FDE_EE)).*

The KEK is a 256 bit derived key material (KBKDF). The DEK is two 256 bit AES-XTS symmetric keys concatenated together and processed as a single 512 bit entity. The DEK is wrapped using the KEK, in accordance with the key wrap defined in Sect. 6.2 of NIST SP 800-38F, and block ciphers as specified in ISO/IEC 18033-3. The key wrap uses 256-bit AES KW using a KBKDF with HMAC-SHA-384 as pseudorandom function. The wrapped key is 256 bits with the security strength of 256. The auth factors are read by the AA component of the TOE stored on either a KTD or a cRSM, indexed by the serial number of the RSMs.

Each RSM has a unique auth factor[5]. Validation is the process of reading the auth factor and comparing what the auth factor thinks is the KEK to what the EE component of the TOE thinks is the KEK. If they match, validation is successful and the resulting comparison yields the auth factor's unwrapped DEK which

[4] Refer to Section 7 for a definition of cryptographically secure random bytes.

[5] *[FDE_AA] Line 7&8, definition of a submask: "The TOE may accept any number of authorization factors, and these are categorized as 'submasks'".*

is then stored in the SATA Controller registers. If validation is unsuccessful, a new auth factor must be generated for that specific RSM. For this evaluation the auth factor is the BEV and is the submask.

The EE component obtains a KEK working down the EE side key chain. The starting point are values stored in the non-volatile memory of the TSEM Flash (TIK[TMK] and Salt) and RSM EEPROM (QEE). The EE derives and unwraps a series of keys to result in a KEK. EE uses this KEK as an argument to TSEM Crypto Library's `AES_unwrap_key()` function with the auth factor as the wrapped key argument (`DEK = AES_unwrap_key( KEK`^{TSEM}`, KEK`^{KTD}`[DEK]`). If the function is successful, the auth factor is validated. If an error is returned, validation fails.

Validation is initiated when:

- at power on or exiting a compliant power saving state, and the IECDL includes both an inserted RSM and either an inserted KTD or inserted cRSM, and
- If a RSM is inserted and a KTD or cRSM is inserted, and the Administrator selects the "Initialize RSM" or "Reinitialize RSM" GUI selections. This selection will reprovision the drive (recreating the values stored in non-volatile memory that derive a KEK) and create a new auth factor for the RSM. Once created, the AA portion of the TOE will store that new auth factor on the KTD, overwriting the previous auth factor.

Otherwise, the validation of the authentication factor is not initiated.

Validation always occurs when an auth factor is read from a KTD or cRSM and all auth factors are validated.

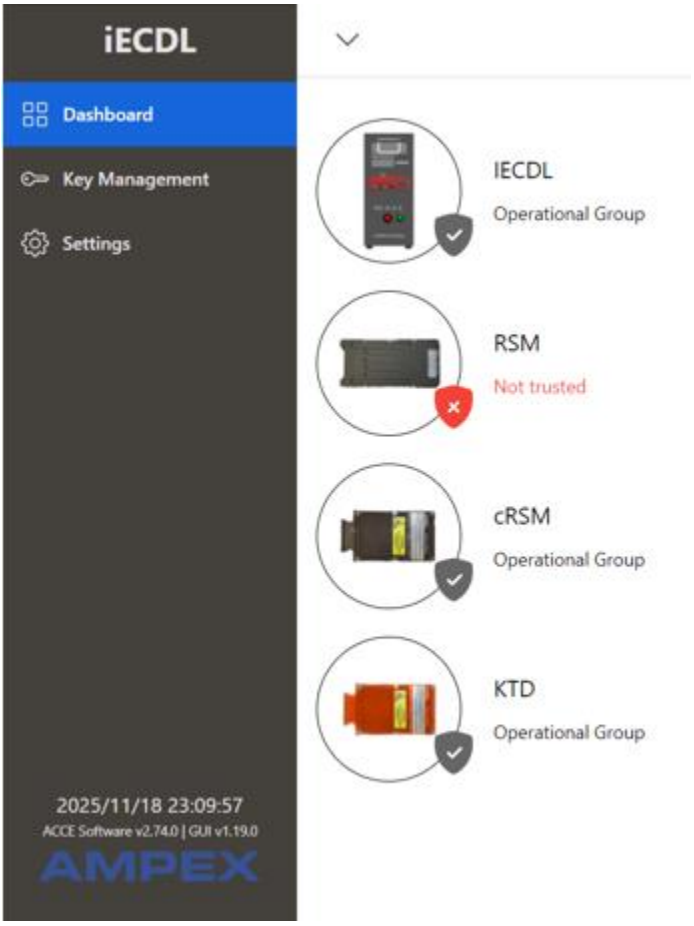
Once validation has been initiated, the AA component of the TOE will attempt to read the KTD, cRSM, or both for the authentication factor, indexed by the RSM's serial number.

There are two places where validation can fail:

- the auth factor associated with the RSM's serial number does not exist on the KTD or cRSM. (This failure occurs in the AA component of the TOE.) or
- the validation process failed (performed in the EE component of the TOE and described above).

If validation fails, the RSM's display in the GUI will be marked as Not Trusted and the RSM's shield icon will be red and access to the TSF data will be blocked.

Figure 3: Authentication Failed Results



When validation is initiated, the AA component will try once on each USB device to find the correct auth factor. If not found, the RSM's display in the GUI will be marked as Not Trusted and the RSM's shield icon will be red. If found, the EE is passed the auth factor (the BEV), the auth factor is validated.

Once the auth factor is received, the EE will validate the auth factor. If the function is successful, the resulting DEK is loaded into the SATA Controller registers. If unsuccessful the EE will try three times to authenticate an auth factor. If only one USB device (KTD or cRSM) is inserted, the EE component will attempt to validate with the auth factor from that device. If authentication fails, the EE will return failure to the AA component and the AA will search again for the auth factor on the USB device. Once found, the auth factor is passed to the EE. If validation fails (it will), the EE will return failure to the AA component. The AA will search again for the auth factor. Once found, the auth factor is passed to the EE. If validation fails, the EE stops validation and marks the RSM as Not Trusted which is visible from the GUI.

If a KTD is inserted and a cRSM is inserted, the AA will search the KTD for the correct auth factor. If found, the AA will pass the auth factor to the EE component of the TOE. If validation fails for the KTD's auth factor, an error will be returned to the AA. The AA will then search the cRSM for an auth factor. If the auth factor is not found on the cRSM, the AA component will search the KTD again. If found, the AA will pass the auth factor to the EE component of the TOE. If authentication fails for the cRSM's auth factor an error will be returned to the AA. The AA will then search the KTD again for an auth factor. The AA will pass the auth factor to the EE component of the TOE. Upon three validation failures, the EE component will stop validation and mark the RSM as Not Trusted which is visible from the GUI.

The auth factor or the state of the iECDL does not affect the ability of a system to enter a Compliant power saving state. If a system loses power, by any means, the system will enter into the G3 state. Otherwise, an Administrator will issue the “shutdown” GUI command to initiate a controlled shutdown into the G2(S5) state. In order to issue this command, the RSMs do not need to be provisioned, keys don’t need to be loaded, and a KTD, cRSM, or RSM does not need to be inserted.

A full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

6.2 User Data Protection (FDP)

6.2.1 FDP_DSK_EXT.1 Protection of Data on Disk (FDE_EE)

The TSEM TOE component includes two FPGA SATA Controllers which are connected to the P1 connector through a SATA interface. The SATA controllers encrypt and decrypt all user data to and from the drives using 256-bit AES in XTS mode of operation provided by the XTS Crypto Library (included in the TOE boundary).

The TOE only encrypts and decrypts user data on the RSM drives. By definition, user data, for this evaluation, excludes the drives metadata. This includes metadata that is maintained per disk block, file information and metadata containing information regarding access to the drive and how the drive is partitioned. The RSMs contains user data. The iECDL includes an internal drive that contains system files such as boot loaders and master boot records that is not encrypted. The iECDL can boot successfully if there isn’t an RSM inserted.

The SATA controllers intermediate all data flows between the TSEM’s NXP Kinetis K81 SoC (System on Chip) and the drives attached to the TOE. There is no direct connection between the TSEM SoC and the SATA interface. The SoC only access the P1 connector directly for USB. This ensures that all user data to and from the drives passes through the SATA controllers and are encrypted and decrypted.

The encryption and decryption of user data stored on the drives is always enabled in the CC evaluated configuration. This is done before the iECDL is delivered and therefore requires no Administrator action.

Data is initially encrypted when a drive is originally provisioned and DEKs are loaded. Any Administrator or User attempt to access the drive’s data is first checked if the Administrator/User has permission to access the data. This layer of permission/role based access is implemented and enforced by the RHEL Operating System, provided by the Operational Environment. Data is decrypted when accessed and encrypted when the data is released. It is transparent to the Administrator/User that the RSM is performing encryption and decryption.

6.3 Security Management (FMT)

6.3.1 FMT_MOF.1 Management of Functions Behavior (FDE_AA)

The TOE’s supported Compliant power saving states do not require management. The system is delivered supporting two Compliant power saving states.

- G2(S5) a soft off state. This state is entered following an Administrator issuing a shutdown command from the GUI interface.

- G3 a mechanical off state. This state is entered when the administrator turns (pushes the off the system or the system loses physical power.

The Host System's RHEL Operating System (Operational Environment) has the OS' non-Compliant power saving states: `sleep, suspend, hibernate, hybrid-sleep, and suspend-then-hibernate` disabled. If an Administrator enables any of these states, the TOE is no longer configured in the CC evaluated configuration. The CC User Guidance contains warnings about tampering with the default system's configuration.

Remote Administrators are warned in the Common Criteria guidance (AGD_CC) to not leave their remote workstations in a lock screen state or a managed power state such as "hibernate or sleep". Additionally, the Administrator is warned to not leave the iECDL unattended until all volatile memory is cleared after a power-off.

6.3.2 Specification of Management Functions

The cPPs require the TOE to support the following list of administrative management commands.

- Change the DEK,
- Erase the DEK,
- Change the Authentication Factor,
- View the current version of the running software,
- View the current version of the running firmware, AND
- Shutdown the system.

The following table maps the cPP mandatory administrative management commands (column three) to the iECDL GUI Commands (column one).

Table 9: iECDL GUI Commands to Mandatory cPP Admin Commands

iECDL GUI Admin Command or Action	Definition	cPP Mandatory Admin Command
Start or connect to the iECDL GUI by any means.	The login page displays the ACCE version supporting the mandatory Admin Command to view the current version of the running software.	View the current version of the running software.
"getAllVersions" iECDL CLI Command	Displays the firmware version supporting the mandatory Admin Command to view the current version of the running firmware.	View the current version of the running firmware.
"Initialize RSM" iECDL GUI Command	Repeats the Provision and Load DEK. Also supports the mandatory "Change Authentication Factor" SFR Admin command.	Change the Authentication Factor
"Initialize RSM or Reinitialize" iECDL GUI Command	Repeats the Provision and Load DEK. Also supports the mandatory "Change the DEK" SFR Admin command and the mandatory "Change Authentication Factor" SFR Admin command.	Change the DEK, Change the Authentication Factor
"Shutdown" GUI Command	Shutdowns the iECDL.	N/A
"Unprovision iECDL" GUI Command	Supports the mandatory "Erase the DEK" SFR Admin command.	Erase the DEK

6.3.2.1 SMF.1/AA Specification of Management Functions (FDE_AA)

The ACCE communicates with the TSEM firmware via a USB connection. RHEL includes a `usbutils` package that enables the ACCE to communicate across the ACCE-to-TSEM USB connection. The ACCE software includes a library, the TSEM Library, that contains the APIs to communicate with the EE component. Refer to Figure 2: iECDL TOE Components for detail.

The AA component includes the software that provides the iECDL GUI and iECDL CLI. It is the link between the Administrator and the TSEM.

The AA component of the TOE implements the following management functions:

a) Forward requests to change the DEK to the EE.

   The DEK can be changed as the result of an Administrator invoking the iECDL GUI "Reinitialize RSM" selection. "Reinitialize RSM" is offered when the RSM is provisioned. Upon invoking the command, the AA component of the TOE will forward the request to the EE component across the USB connection.

b) Forward requests to cryptographically erase the DEK to the EE.

   The DEK can be erased as the result of an Administrator selecting the "Unprovision iECDL" iECDL GUI option. Upon invoking the command, the AA component of the TOE will forward the request to the EE component across the USB connection.

c) Allow authorized uses to change authorization factors or set of authorization factors used within the supported authorization method.

   The authorization factor can be changed as the result of an Administrator invoking the iECDL GUI "Initialize RSM" or "Reinitialize RSM" command. Upon invoking the command, the AA component of the TOE will forward the request to the EE component across the USB connection.

d) Support the user initiating a TOE software update. This action updates the ACCE software, the AA component.

   Refer to Section FPT_TUD_EXT.1/AA Trusted Update *(FDE_AA)* in this document.

Refer to the developer's proprietary annex *Key Management Description* associated with this ST for a detail description of the TOE implementing the above commands. The annex is made available at the discretion of the developer.

6.3.2.2 FMT_SMF.1/EE Specification of Management Functions *(FDE_EE)*

The EE component of the TOE implements the following mandatory management functions:

a) Change the DEK, as specified in FCS_CKM.1(b) Cryptographic Key Generation (Symmetric Keys) *(FDE_EE)*, when re-provisioning or when commanded.

   The "Reinitialize RSM" command will be received by the TSEM from the Host System across the USB connection. Once received, the EE component securely erases[6] the RSM data, generates a new KEK, generates new AES-XTS key pairs (DEKs), and saves the DEKs in the SATA Controller registers (overwrite).

[6] Refer to Section 7 for a definition of secure erase.

The EE then encrypts the DEKs (wraps with the new KEK), and passes the wrapped DEK (auth factor) to the Host System via the USB connection. The Host System will then fetch the serial number of the RSM, search the KTD for that serial number, and overwrite the auth factor with the new auth factor.

This command leaves the drives readable/writeable, however, all the data on the drive before the command was issued, is lost.

b) Erase the DEK, as specified in FCS_CKM.4(a)/EE Crypto Key Destruction (Power Management) *(FDE_EE).*

The "Unprovision iECDL" command will be received by the TSEM across the USB connection. The command erases (overwrites with zeroes) the DEKs held in the SATA Controller registers plus overwrites with zeroes the key material held in TSEM's non-volatile memory and held in the RSM's non-volatile memory used to validate the auth factor. This command leaves the drives unreadable and unwritable and all encrypted data is lost. An Administrator must issue the "Initialize RSM" command to get the system in reconfigured.

c) Initiate a TOE firmware update.

An Administrator can initiate a firmware update as described in FPT_TUD_EXT.1/EE TSS below.

Refer to the developer's proprietary annex *Key Management Description* associated with this ST for a detail description of the TOE implementing the above commands. The annex is made available at the discretion of the developer.

6.3.3 FMT_SMR.1 Security Roles (FDE_AA)

There are no TSS evaluation activities for this SFR.

The AA component of the TOE supports two Administrator interfaces: the iECDL GUI and iECDL CLI. Both interfaces require the Administrator to enter a username and a password. There is one Administrator role supported by the iECDL: Security Administrator which is root[7] or a user with elevated superuser[8] privileges. User creation and roles is handled by the RHEL Operating System (Operational Environment) and outside the scope of this evaluation.

6.4 Protection of the TSF (FPT)

6.4.1 FPT_KYP_EXT.1/AA Protection of Key and Key Material (FDE_AA)

The AA component of the TOE does not implement submask combining (FCS_SMC_EXT.1 Submask Combining *(FDE_AA)).*

The AA component of the TOE's key chain length is one. The auth factor, read from the KTD or cRSM is passed to the TSEM as the BEV. The AA component does not create or derive any intermediate keys.

Table 14 and Table 15 provides details for storage and protection of the volatile and non-volatile keys respectively.

[7] Refer to Section 7 for a definition of root.

[8] Refer to Section 7 for a definition or superuser.

6.4.2 FPT_KYP_EXT.1/EE Protection of Key and Key Material (FDE_EE)

The EE component of the TOE does not generate intermediate keys using submask combining (FCS_SMC_EXT.1 Submask Combining *(FDE_EE)).*

The EE component of the TOE maintains a key chain of five. It begins with the BEV (the KEK wrapped DEK (KEK[DEK]) auth factor), received from the Host System and ends with a DEK, if the auth factor is successfully validated and ends with an error if a validation error occurs. If the key chain was successfully navigated (validation was successful), DEKs are stored in the SATA Controller registers and the Administrator and Users have access to the data held encrypted on the RSM drives.

The intermediate keys supported by the EE component of the TOE are either derived from another key or created from decrypting (unwrapping) another key. A subset of intermediate keys are created starting from decrypting a wrapped key stored in the non-volatile memory of the TSEM (Flash). The keys that unwrap the keys are derived from intermediate keys in the keychain. Intermediate keys are also created by a Key-Based Key Derivation Function (KBKDF). The key and context values used in the KBKDF function are obtained from the Host System, read from TSEM Flash, and read from RSM EEPROM.

All keys used as input to the keychain are stored encrypted (wrapped) in non-volatile memory of the TSEM. All intermediary keys derived or decrypted during navigation of the keychain are overwritten with zeroes immediately after use.

The specifics of the iECDL key chain is consider developer proprietary however, a full description of the cryptographic keys and protection thereof is provided in a developer proprietary annex *Key Management Description* associated with this ST. The annex is made available at the discretion of the developer.

Additionally, the keys, their origin, and their use is described in Section 6.6 Cryptographic Key Destruction in this document.

6.4.3 FPT_PWR_EXT.1/AA Power Saving States (FDE_AA) *and* FPT_PWR_EXT.1/EE Power Saving States (FDE_EE)

The TOE supports two Compliant power saving states:

- G2(S5) – Soft off state. The soft off state is when the system fully shuts down without a hibernation file. Soft off is also known as a full shutdown. During a full shutdown and boot, the entire user session is torn down and restarted on the next boot. Consequently, a boot/startup from this state takes significantly longer than S1-S4[9]. A full shutdown (S5) occurs when a system shut down is requested or when an application calls a shutdown API.
- G3 – Mechanical off state. In this state, the system is completely off and consumes no power. The system returns to the working state only after a full reboot.

6.4.4 FPT_PWR_EXT.2/AA Timing of Power Saving States (FDE_AA) and FPT_PWR_EXT.2/EE Timing of Power Saving States (FDE_EE)

The TOE's Compliant power saving states can be entered as follows.

- G2(S5) is entered following an Administrator issuing a "shutdown" command from the iECDL GUI interfaces.

[9] Refer to Appendix A for a definition of System Power States.

- G3 is entered when the administrator pushes the iECDL's front panel POWER button or the iECDL loses physical power.

The Host System's RHEL Operating System (Operational Environment) has the OS' non-Compliant power saving states: `sleep, suspend, hibernate, hybrid-sleep, and suspend-then-hibernate` disabled. If an Administrator enables any of these states, the TOE is no longer configured in the CC evaluated configuration. The CC User Guidance (AGD_CC) contains warnings about tampering with the default system's configuration.

The iECDL's POWER button is on its front panel. Because shutting down the iECDL using the POWER button does not initiate a graceful system shutdown[10], selecting the POWER button to enter into a Compliant power saving state is not recommended. CC User guidance (AGD_CC) includes a warning statement stating that the POWER button is does not initiate a graceful system shutdown and as a result, the "shutdown" GUI command is recommended.

Remote Administrators are warned in the Common Criteria guidance (AGD_CC) to not leave their remote workstations in a lock screen state or a managed power state such as "hibernate or sleep". Additionally, the Administrator is warned to not leave the iECDL unattended until all volatile memory is cleared after a power-off. This is indicated in the iECDL's front panel LEDs Power LED. The LED will be a solid green when the iECDL is on and not lit when the iECDL is off.

6.4.5 FPT_TST_EXT.1/AA TSF Testing *(FDE_AA)*

The AA component's *AMPEX OpenSSL Cryptographic Module* runs a group of self-tests at power-up referred to as FIPS Power-On Self Tests. This test suite includes Cryptographic Algorithm Self-Tests (CASTs) on all approved cryptographic algorithms.

Table 10: Cryptographic Algorithm Self-Tests

Algorithm	Test Properties	Test Method	Test Type	Details
RSA SigVer FIPS186-4)	3072-bit key	Signature Verification	KAT (Known Answer Tests)	Signature verification using RSA.
SHA-384	384-bit digest	Message digest	KAT (Known Answer Tests)	Hash function message digest used as part of integrity and signature service.

The system performs self-tests on all approved cryptographic algorithms that are part of the approved services supported in the approved mode of operation. The tests type run is CASTs using the KAT (Known Answer Tests) method. The tests run, include but are not limited to, the algorithms identified Section 6.5 CAVP Algorithm Certificate Details. During these tests, services are not available, and data output (via the data output interface) is inhibited during the conditional self-tests. If any of these tests fail, the system transitions to the Error State. The TOE does not have any non-cryptographic functions which would affect the correct operation of the TSF. All non-cryptographic functions are covered under the FIPS Power-On Self Tests performed by the AMPEX OpenSSL Cryptographic Module.

[10] Refer to Section 7 for a definition of.

6.4.6 FPT_TST_EXT.1/EE TSF Testing *(FDE_EE)*

When the TOE is powered on and the TSEM starts execution of the TOE Firmware, the TOE firmware enters an initialization state for basic initialization of the TOE. Upon completion of the basic initialization, the TOE enters a self-testing state where a sequence of self-tests are executed in a precise order. Each self-test is a state from which the TOE firmware either transitions to the next self-test state, to the provisioning or operational state (in case of the last self-test being successfully completed), or to an error state if the self-test fails.

The self-tests the TOE executes are various Cryptographic Algorithm Self Tests (CAST) and the verification of the digital signature of the TOE firmware. They are (in the order of execution) the following:

1. ECDSA signature verification CAST, inclusive of SHA-384.
2. Firmware integrity test (i.e. verification of the firmware digital signature).
3. KBKDF key derivation CAST, inclusive of HMAC and SHA-384.
4. Separate key wrap and unwrap CASTs, inclusive of forward AES cipher.
5. Invocation of TSEM Cryptographic Library DRBG initialization function with automatic self-tests. The self-tests include all DRBG health tests applicable to Hash DRBG implemented with SHA-256 stated in Sect. 11 of NIST SP 800-90A.

Once completed the FPGA perform the self-tests. These include separate AES-XTS encrypt and decrypt tests performed on each Lycan instance and are inclusive of embedded AES forward and inverse ciphers.

The TOE does not have any non-cryptographic functions which would affect the correct operation of the TSF. All non-cryptographic functions are covered under the firmware integrity test performed by TSEM.

6.4.7 FPT_TUD_EXT.1/AA Trusted Update (FDE_AA)

iECDL Administrators will be notified by email if a new version of the ACCE has been released. The email will also contain the address of a secure site where the new version can be downloaded. The new version can be copied to the iECDL by a remote user using SCP (Secure Copy Protocol). Details of updating the AA software, ACCE, are available in the CC user guidance (AGD_CC).

The iECDL ACCE version number can be viewed by the iECDL GUI. To do this, invoke the iECDL GUI. The iECDL login screen will appear. On the login page, the current time and date will be displayed along with the current version of the ACCE Software. e.g. “ACCE Software v2.74.1 | GUI v1.19.0”. Ensure you are running the version listed in Section 1.5.2 of this document under “Ampex Software Build”. Or, if an upgrade has been performed, ensure the ACCE Software version displayed matches the version number received in your Ampex update email. If they do not match, contact the Ampex support listed in the CC guidance, (AGD_CC).

The TOE uses RPM (Red Hat Package Manager) to update the ACCE software. RPMs are signed with an Ampex signing key which is an RSA 3072-bit key. When a package is installed, the signature on the package is validated by the RPM tool (Operational Environment). The RPM file includes digests of the files within the RPM. These digests are stored in a database on the system during package install. During TOE upgrade, the contents of each file are verified against the stored digests.

Administrative access to initiate RPM is by using the RHEL CLI. Administrators stop the running ACCE software, uninstall the ACCE, install the new version, and reboot the system. An error will be displayed if the digital signature verification failed. Refer the CC Guidance (FDE_CC) for specific instructions.

The protection and maintenance of the TOE update credentials is provided by the Operational Environment.

6.4.8 FPT_TUD_EXT.1/EE Trusted Update *(FDE_EE)*

iECDL Administrators will be notified by email if a new version of the TSEM firmware has been released. The email will also contain the address of a secure site where the new version can be downloaded. The new version can be copied to the iECDL by a remote user using SCP (Secure Copy Protocol). Details of updating the EE firmware are available in the CC user guidance (AGD_CC).

The TSEM Firmware version number can be viewed by the iECDL CLI. To do this, the Administrator logs on to the iECDL using the iECDL CLI. Then the command `getAllVersions` is entered. The TSEM firmware version will be displayed under the `TSEM FW =` entry. The Administrator must ensure that the value displayed matches the version listed in Section 1.5.2 of this document under "TSEM Firmware". Or, if an upgrade has been performed, the Administrator must ensure the TSEM Firmware version displayed matches the version number received in the Ampex update email. If they do not match, the Administrator should contact Ampex support listed in the CC guidance, (AGD_CC). The TSEM Firmware version number can also be viewed by the TSEM CLI via the command `-v --command version.`

To upgrade the firmware, the Administrator must first download a developer's version of the ACCE software and upgrade the current ACCE with the developer's version (perform the FPT_TUD_EXT.1/AA Trusted Update (FDE_AA) upgrade described above). The developer's version includes a CLI, the TSEM CLI, that includes the Administrator commands that enable the firmware to be updated. Updating the TSEM firmware involves disassembling the iECDL, connecting cables to the TSEM card, and downloading the new firmware. This is a complicated process and reassembling the iECDL correctly is complicated. Therefore, customers are encouraged to send their iECDL to Ampex when new versions of the firmware are released. However, detailed instructions are provided in the CC User Guidance (AGD_CC).

The TSEM firmware is signed with an Ampex signing key which is an ECDSA P-384 with SHA2-384 key that meets FIPS PUB 186-4 standards and meets ISO/IEC 14888-3, Section 6.4, for ECDSA schemes. When the firmware is installed, the signature on the package is validated by the TSEM using the TSEM Crypto Library (inside the TOE boundary).

The signing key is a long-term private key known only to the Ampex production environment of the TOE. The production environment is the only party in possession of the signing key and the only authorized source of the firmware upgrades. The public key corresponding to the private key of the production environment is stored in the non-volatile memory (Flash) of the SoC of the TSEM during the production of the TOE (FW_Sig_Key). Consequently, any firmware upgrade whose digital signature can be successfully verified originates from an authorized source.

Refer the CC Guidance (FDE_CC) for specific instructions of updating the TSEM firmware.

6.4.9 FPT_FUA_EXT.1 Firmware Update Authentication (FDE_EE)

The TSEM firmware is signed with an Ampex signing key which is an ECDSA P-384 with SHA2-384 key that meets FIPS PUB 186-4 standards and meets ISO/IEC 14888-3, Section 6.4, for ECDSA schemes. When the firmware is installed, the signature on the package is validated by the TSEM using the TSEM Crypto Library (inside the TOE boundary).

To upgrade the firmware, the Administrator must first download a developer's version of the ACCE software and upgrade the current ACCE with the developer's version (perform the FPT_TUD_EXT.1/AA Trusted Update (FDE_AA) upgrade described above). The developer's version includes a CLI, the TSEM CLI,

that includes the Administrator commands that enable the firmware to be updated. Updating the TSEM firmware involves disassembling the iECDL, connecting cables to the TSEM card, and downloading the new firmware. Detailed instructions are provided in the CC User Guidance (AGD_CC).

The firmware is burned into Flash of the TSEM.

The signing key is a long-term private key known only to the Ampex production environment of the TOE. The production environment is the only party in possession of the signing key and the only authorized source of the firmware upgrades. The public key corresponding to the private key of the production environment is stored in the non-volatile memory (Flash) of the SoC of the TSEM during the production of the TOE (FW_Sig_Key). Consequently, any firmware upgrade whose digital signature can be successfully verified originates from an authorized source.

The digital signature of the firmware is validated on each system boot. If the digital signature validation fails, an error will be written to the audit/log file and the iECDL will not complete the reboot and enter into an error state. To recover from this state, a valid firmware version needs to be uploaded.

6.5 CAVP Algorithm Certificate Details

The following table identifies the cryptographic libraries used by the iECDL. The AA component of the TOE, ACCE, uses the AMPEX OpenSSL Cryptographic Module, version 1.1.1. This library is included in the TOE boundary. The EE component of the TOE uses two cryptographic libraries: the TSEM Cryptographic Library v1.1.16.0 and the SATA Crypto Library version EP2AGX65. The TSEM Cryptographic Library is used by the TSEM software. The SATA Crypto Library is used on the FPGA to AES-XTS encrypt and decrypt the RSM data. Both libraries are included in the TOE boundary.

The following table identifies the three cryptographic libraries used by the iECDL. The next table identifies the cryptographic algorithms used by the iECDL.

Table 11: iECDL Cryptographic Library CAVP Details

CAVP Cert Implementation Name	Runs On	CAVP Cert Operating Environment	CAVP Cert Description	CAVP Cert Version	CAVP Cert Vendor	CAVP Cert Number
AMPEX OpenSSL Cryptographic Module	Host System Card	RHEL 8.6 on AMPEX TuffServ with Intel® Atom® C3558 (Goldmont microarchitecture)	The AMPEX OpenSSL Cryptographic Module is a security module that provides cryptographic functions and services to a wide range of AMPEX products.	1.1.1	Ampex Data Systems Corporation	#A7741
TSEM Cryptographic Library	TSEM	NXP K81 (ARM Cortex M4)	The TSEM Cryptographic Library provides the cryptographic primitives required for the TSEM	1.1.16.0	Ampex Data Systems Corporation	#A2921

CAVP Cert Implementation Name	Runs On	CAVP Cert Operating Environment	CAVP Cert Description	CAVP Cert Version	CAVP Cert Vendor	CAVP Cert Number
			Cryptographic Module, leveraging K81 hardware acceleration.			
IPC-BL120B-ZM	TSEM	IPC-BL120B-ZM	The IntelliProp AES-XTS Encryption IP core.	EP2AGX65	Ampex Data Systems Corporation	#A2914

Table 12: iECDL TSEM CAVP Algorithm Table

Algorithm	Standard	Modes Supported	Library	CAVP Certificate
Cryptographic Operation (Signature Verification) *(FDE_AA)* (FCS_COP.1(a)/AA)				
RSA SigVer (FIPS186-4)	FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1-v1 5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3, for RSA schemes.	3072 bits	AMPEX OpenSSL Cryptographic Module	#A7741
Cryptographic Operation (Signature Verification) *(FDE_EE)* (FCS_COP.1(a)/EE)				
ECDSA SigVer (FIPS186-4)	FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves"; ISO/IEC 14888-3, Section 6.4, for ECDSA schemes.	P-384 SHA2-384	TSEM Cryptographic Library	A2921 (NXP K81)
Cryptographic Operation (Hash Algorithm) *(FDE_AA)* (FCS_COP.1(b)/AA)				
SHA2-384	ISO/IEC 10118-3:2004	SHA-384	AMPEX OpenSSL Cryptographic Module	#A7741
Cryptographic Operation (Hash Algorithm) *(FDE_EE)* (FCS_COP.1(b)/EE)				

Algorithm	Standard	Modes Supported	Library	CAVP Certificate
SHA2-256	ISO/IEC 10118-3:2004	SHA-256	TSEM Cryptographic Library	A2921 (NXP K81)
SHA2-384	ISO/IEC 10118-3:2004	SHA-384	TSEM Cryptographic Library	A2921 (NXP K81)
Cryptographic Operation (Keyed Hash Algorithm) *(FDE_EE)* (FCS_COP.1(c))				
HMAC-SHA2-384	ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2"	256 bits key length MAC: 256	TSEM Cryptographic Library	A2921 (NXP K81)
Cryptographic Operation (Key Wrapping) *(FDE_EE)* (FCS_COP.1(d))				
AES-KW	AES: ISO/IEC 18033-3 KW: NIST SP 800-38F	Key Length 256	TSEM Cryptographic Library	A2921 (NXP K81)
Cryptographic Operation (AES Data Encryption/Decryption) *(FDE_EE)* (FCS_COP.1(f))				
AES-XTS Testing Revision 2.0	AES: ISO/IEC 18033-3 XTS: IEEE 1619	256 bits key length Payload Length: 4096	IPC-BL120B-ZM	A2914
Cryptographic Key Derivation *(FDE_EE)* (FCS_KDF_EXT.1)				
KDF SP800-108	NIST SP 800-108 (KDF in Counter Mode)	Counter Mode HMAC-SHA2-384 MAC Mode 256 bits key length	TSEM Cryptographic Library	A2921 (NXP K81)
Random Bit Generation *(FDE_EE)* (FCS_RBG_EXT.1)				
Hash_DRBG(any)	NIST SP 800-90A, ISO/IEC 18031:2011	SHA2-256	TSEM Cryptographic Library	A2921 (NXP K81)

6.6 Cryptographic Key Destruction

The tables below describes the key zeroization provided by the TOE and as referenced in FCS_CKM.4(b) Cryptographic Key Destruction (TOE-Controlled Hardware) *(FDE_EE)* and FCS_CKM.4(d) Cryptographic Key Destruction (Software TOE, 3rd Party Storage) *(FDE_AA)* which describes the characteristics of cryptographic keys and key material.

Refer to the developer's proprietary annex *Key Management Description* associated with this ST for a detail description of the TOE implementing the above commands. The annex is made available at the discretion of the developer.

The following three tables are considered an extension of the TSS.

The following table identifies the keys and keying material used by the TOE.

Key

Key Material (KM) does not have to be protected.
Red font indicates keys.
Green font indicates Key Material (KM).

6.6.1 Keys and Key Material

Note: refer to Appendix A, Section 3, for a description of data in TPM, Flash, and EEPROM.

Table 13: Keys and Key Material

The Key	Key/KM	Purpose and Details	How the Key is Protected	How the Key/KM is Created/Derived/Unwrapped	Strength of Key
BEV (Border Encryption Value)	KM	BEV is a term used in the FDE cPPs to identify the key material passed from the AA to the EE. In this evaluation the BEV is the KEK[DEK].	N/A	N/A	N/A
DEK (Data Encrypting Key)	Key	Used to encrypt/decrypt the RSM data in AES-XTS mode. AES-XTS uses two 256-bit symmetric keys. The two keys are stored as a single encrypted 512-bit entity in the SATA Controllers.	Stored encrypted (wrapped) when stored on the KTDs (KEK[DEK]). Protected when stored in the SATA Controller registers by the Controllers implementing RAP. DEK	A call to the TSEM Crypto Library's `VP_EncryptInit_ext()` function. *DEKs* = `EVP_EncryptInit_ext( AES, 256, XTS)` A call to the TSEM Crypto Library's unwrap function, `aes-256-unwrap()`. *DEK* = `aes-256-unwrap(KEK, KEK[DEK])`	256 bits
FW_Sig_Key (FW Signature Key)	Key	An ECDSA P-384 with SHA2-384 key stored in TSEM non-volatile memory (Flash). Used by the TSEM at system boot to validate the digital signature of the FW at system boot. It is the public signing key.	Stored protected in the TSEM's non-volatile memory (Flash)[11].	The key is created by the Ampex production team when a new TSEM FW build is created. The key is stored in the TSEM non-volatile memory (Flash) at that time.	256 bits

[11] Refer to section A.3 for a description of how data stored in a Flash non-volatile memory is protected.

The Key	Key/KM	Purpose and Details	How the Key is Protected	How the Key/KM is Created/Derived/Unwrapped	Strength of Key
KEK (Key Encrypting Key)	Key	A 256-bit AES key used for wrapping and therefore, encrypting the DEK, KEK[DEK]. It is an intermediate key in the EE's key chain used to validate the auth factor.	A wrapping key is not considered secure when it's wrapping a key and must be stored in a secure storage. The KEK[DEK] is stored protected in the KTD's Flash, non-volatile memory.	A call to the TSEM Crypto Library's KBKDF function, `EVP_KDF-KBs()`. *KEK* = `EVP_KDF-KB` *(TMK, QEE)*	256 bits
KEK[DEK] (The auth factor, the BEV)	KM	The DEK, wrapped by the KEK is the TOE's Authorization Factor and used to determine if access is allowed to the encrypted data stored in the RSMs. The wrapped key is 256 bits.	Stored protected in the KTD's Flash, non-volatile memory.	A call to the TSEM Crypto Library's wrap function, `aes-256-wrap()`. KEK[DEK] = `aes-256-wrap` (KEK, DEK)	The strength of a wrapped key is the strength of the wrapping key. 256 bits
PIK (Platform Identity Key)	Key	The 256-bit AES PIK is used for generating the TIK (KBKDF derived). *TIK = KBKDF(PIK, Salt)* It is an intermediate key in the EE's key chain used to validate the auth factor.	Stored protected in the Host System's TPM, non-volatile memory[12].	Generated by the TPM's manufacturer and stored in non-volatile memory of the Host System (TPM) before delivery of the iECDL.	256 bits
QEE (Quick Erase Element)	KM	A 256-bit random number used to derive (KBKDF) the KEK. *KEK = KBKDF(TMK, QEE)*	Stored protected in the RSM's EEPROM, non-volatile memory[13].	A call to the TSEM Crypto Library's get random function, `rand()`. *QEE = `rand(32)`*	N/A

[12] Refer to Section 7, Table 16: Terms for a definition of how values held in a TPM are protected.
[13] Refer to section A.3 for a description of how data stored in EEPROM is protected.

The Key	Key/KM	Purpose and Details	How the Key is Protected	How the Key/KM is Created/Derived/Unwrapped	Strength of Key
RSN_SN# (RSN Serial Number)	KM	The serial number of the inserted RSM. Authorization factors (KEK wrapped DEKs, KEK[DEK]) are stored on KTDs indexed by their RSMs serial number.	Stored protected in the RSM's EEPROM, non-volatile memory[14].	Generated by the RSM manufacturer and stored in non-volatile memory of the RSMs (EEPROM) before delivery of the iECDL.	N/A
		Used to index the auth factors stored on a KTD KEK[DEK].	Stored protected in the KTD's Flash, non-volatile memory[15].	When an authfactor is stored on the KTD.	N/A
Salt	KM	256 bits of random data used as context to derive the TIK. *TIK = KBKDF(PIK, Salt)* It is intermediate key material in the EE's key chain used to validate the auth factor.	Stored in the TSEMs non-volatile memory (Flash).	A call to the TSEM Crypto Library's get random function, `rand()`. *.Salt = `rand(32)`*	N/A
TIK (TSEM Identity Key)	Key	A 256-bit AES key used to wrap the TMK when stored in the TSEM's non-volatile memory (Flash). *TIK[TMK] = KW_EN(TIK, TMK)* It is an intermediate key in the EE's key chain used to validate the auth factor.	A wrapping key is not considered secure when it's wrapping a key and must be stored in a secure storage. The TIK[TMK] is stored protected in the TSEM's Flash non-volatile memory.	A call to the TSEM Crypto Library's KBKDF function, `EVP_KDF-KBs()`. *TIK* = `EVP_KDF-KB` *(PIK, Salt)*	256 bits
TIK[TMK] (TIK wrapped TMK)	KM	The TMK wrapped with the TIK. The wrapped key is stored in non-volatile	Stored protected in TSEM's Flash,	A call to the TSEM Crypto Library's wrap function, `aes-256-wrap()`,	256 bits

[14] Refer to section A.3 for a description of how data stored in EEPROM is protected.
[15] Refer to section A.3 for a description of how data stored in Flash is protected.

The Key	Key/KM	Purpose and Details	How the Key is Protected	How the Key/KM is Created/Derived/Unwrapped	Strength of Key
		memory and used as the start of the EE's key chain.	non-volatile memory.	*TIK[TMK]* = `aes-256-wrap(`*TIK, TMK)*	
TMK (TSEM Master Key)	Key	The 256-bit AES TMK is a symmetrical key used for verifying the authenticity of commands received by the TOE and, with QEE as context, to derive KEK. *KEK = KBKDF(TMK, QEE)* It is an intermediate key in the EE's key chain used to validate the auth factor.	Used in RAM only to derive the KEK.	A call to the TSEM Crypto Library's function to create a symmetric key, `rand()`. *TMK* = `rand(32)`. A call to the TSEM Crypto Library's unwrap function, `aes-256-unwrap()`. *TMK* = `aes-256-wrap(`*TIK, TIK[TMK])*	256 bits

6.6.2 The iECDL Keys and Key Material Lifecycle in Volatile Memory

Table 14: The iECDL Keys and Key Material Lifecycle in Volatile Memory

The Key	Key/KM	When the Key/KM is Introduced into the Host System's or TSEM's Volatile Memory.	Use	When the Key/KM is Destroyed	How the Key/KM is Destroyed
BEV (Border Encryption Value)	The BEV is the KEK[DEK] which is the authentication factor and described below.				
DEK (Data Encrypting Key) In TSEM volatile memory.	Key	Created by the TSEM during: • Create_Keys *DEK = create XTS keys*	Used to create the auth factor. $KEK^{new}[DEK^{new}]$ = KW_EN(KEK, DEK) The TSEM sends the auth factor to the Host System.	After use in TSEM volatile memory.	Overwritten with zeroes.
	Key	Derived by the TSEM as the result of a successful validation. • Put_Keys *DEK = KW_DE(KEK^{TSEM}, KEK^{KTD}[DEK]).*	Used by the TSEM to store in the SATA Controller registers.	After use in TSEM volatile memory.	Overwritten with zeroes.

The Key	Key/KM	When the Key/KM is Introduced into the Host System's or TSEM's Volatile Memory.	Use	When the Key/KM is Destroyed	How the Key/KM is Destroyed
DEKs in SATA Controller registers (volatile)	Key	Stored in the SATA Controller registers during: • Put_Keys	Used to encrypt/decrypt the RSM data in AES-XTS mode. DEK	Upon invocation of the "shutdown" command. Upon invocation of the "Unprovision iECTL" GUI Command"[16] Upon the loss of power by any means.	Overwritten with zeroes.
KEK (Key Encrypting Key)	Key	Derived by the TSEM for validation during: • Put_Keys() TSEM function. $KEK^{TSEM} = KBKDF(TMK, QEE)$.	Used to validate the auth factor by using the derived KEK from TSEM to unwrap the auth factor. $DEK = KW\_DE(KEK^{TSEM}, KEK^{KTD}[DEK])$. Also used to derive the DEK if validation was successful.	After use in TSEM volatile memory.	Overwritten with zeroes.
	Key	Derived by the TSEM to create a new auth factor during: • Create_Keys() TSEM function. $KEK = KBKDF(TMK, QEE)$.	Used by the TSEM to create a new auth factor: $KEK^{new}[DEK^{new}] = KW\_EN(KEK, DEK)$ The TSEM passes the value to the Host System to store on the KTD.	After use in TSEM volatile memory.	Overwritten with zeroes.
KEK[DEK] (The auth factor, the BEV)	KM	Read by the Host System from the KMD during: • Put_Keys() TSEM function	Used as the first step in validation. The Host System reads the auth	After use in Host System volatile memory.	Overwritten with zeroes.

[16] The "Unprovision iECDL" GUI command supports the "Erase DEK" FMT_SMF.1/AA and FMT_SMF.1/EE mandatory management command. Upon invocation, the TSEM calls the destroy() TSEM Command (refer to the KMD).

The Key	Key/KM	When the Key/KM is Introduced into the Host System's or TSEM's Volatile Memory.	Use	When the Key/KM is Destroyed	How the Key/KM is Destroyed
		and passed to the TSEM.	factor KEK[DEK] from the KTD and sends it to the TSEM.		
	KM	Received from the Host System during: • Put_Keys() TSEM function.	Used as the second step in the validation process. The TSEM receives the KEK[DEK] and validates it by using the TSEM derived KEK to unwrap the auth factor (KEK[DEK]). *DEK = KW_DE(KEK^{TSEM}, KEK^{KTD}[DEK]).* If the DEK is returned, validation was successful. If an error is returned, validation failed.	After use in TSEM volatile memory.	Overwritten with zeroes.
			Also used to derive the DEK if validation was successful if validation is successful.		
	KM	Created by the TSEM during: • Create_Keys() TSEM function and passed to the Host System.	Created by the TSEM to create a new auth factor: KEK^{new}[DEK^{new}] = KW_EN(KEK, DEK) The TSEM passes the value to the Host System to store on the KTD.	After use in TSEM volatile memory.	Overwritten with zeroes.
	KM	Received by the Host System from the TSEM during: • Create_Keys() TSEM function. KEK^{new}[DEK^{new}]	Received by the Host System from the TSEM to store the new auth factor in KTD. KEK^{new}[DEK^{new}]	After use in Host System volatile memory.	Overwritten with zeroes.

The Key	Key/KM	When the Key/KM is Introduced into the Host System's or TSEM's Volatile Memory.	Use	When the Key/KM is Destroyed	How the Key/KM is Destroyed
PIK (Platform Identity Key)	Key Symmetric 256-bits	Read from the TPM by the Host System and passed to the TSEM during: • Provision() TSEM function, • Put_Keys() TSEM function, and • Create_Keys() TSEM function.	Used to pass to TSEM for use in the three functions.	After use in Host System volatile memory.	Overwritten with zeroes.
		Received by the TSEM from the Host System during: • Provision() TSEM function, • Put_Keys() TSEM function, and • Create_Keys() TSEM function.	Used by the TSEM to derive the TIK *TIK = KBKDF(PIK, Salt)* for the use in the three functions.	After use in TSEM volatile memory.	Overwritten with zeroes.
QEE(Quick Erase Element)	KM random number 256 bits	Created by the TSEM and stored in the RSM's non-volatile memory (EEPROM) during: • Provision() TSEM function.	Stored in the RSM's non-volatile memory (EEPROM) for future use by the TSEM to derive a KEK. *KEK = KBKDF(TMK, QEE).*	After use in TSEM volatile memory.	Overwritten with zeroes.
		Read by the Host System from the RSM's EEPROM and passed to the TSEM during: • Put_Keys() TSEM function and • Create_Keys() TSEM function.	Used to pass to TSEM for use in the two functions.	After use in Host System volatile memory.	Overwritten with zeroes.
		Received by the TSEM from the Host System during: • Put_Keys() TSEM function and • Create_Keys() TSEM function.	Used to derive the KEK to use in validation of the authentication factor (KEK[DEK]). *KEK = KBKDF(TMK, QEE).*	After use in TSEM volatile memory.	Overwritten with zeroes.
RSN_SN# (RSN Serial Number) QEE		Read by the Host System from the RSM's EEPROM during:	Used to search the KTD for the correct entry to store the	After use in Host System	Overwritten with zeroes.

The Key	Key/KM	When the Key/KM is Introduced into the Host System's or TSEM's Volatile Memory.	Use	When the Key/KM is Destroyed	How the Key/KM is Destroyed
(Quick Erase Element)		• Create_Keys() TSEM command.	new KEK[DEK] In the KTD.	volatile memory.	
		Read by the Host System from the RSM's EEPROM during: • Put_Keys() TSEM command.	Used to search the KTD for the correct entry to retrieve the stored KEK[DEK] In the KTD for validation. The KEK[DEK] is then passed to the TSEM.	After use in Host System volatile memory.	Overwritten with zeroes.
Salt	KM	Created by the TSEM and stored in TSEM's non-volatile memory (Flash) during: • Provision() TSEM function.	Stored in TSEM non-volatile memory (Flash) for future use in deriving the TIK. *TIK = KBKDF(PIK, Salt)*	After use in TSEM volatile memory.	Overwritten with zeroes.
	KM	Read by the TSEM from the TSEM's non-volatile memory during: • Put_Keys() TSEM function and • Create_Keys() TSEM function.	Used to derive the TIK. *TIK = KBKDF(PIK, Salt)*	After use in TSEM volatile memory.	Overwritten with zeroes.
TIK (TSEM Identity Key)	Key	Derived by the TSEM during: • Provision() TSEM command. *TIK = KBKDF(PIK, Salt)*	Used to wrap the TMK. *TIK[TMK] = KW_EN(TIK, TMK)* and store the value in TSEM non-volatile memory (Flash) for future use when validating the auth factor.	After use in TSEM volatile memory.	Overwritten with zeroes.
		Derived by the TSEM during • Put_Keys() TSEM function and • Create_Keys() TSEM function.	Used to unwrap the TMK. *TMK = KW_DE(TIK, TIK[TMK])*	After use in TSEM volatile memory.	Overwritten with zeroes.

The Key	Key/KM	When the Key/KM is Introduced into the Host System's or TSEM's Volatile Memory.	Use	When the Key/KM is Destroyed	How the Key/KM is Destroyed
		TIK = KBKDF(PIK, Salt)			
TIK[TMK] (TIK wrapped TMK)	SE	Created (wrapped) by the TSEM during: • Provision() TSEM command. *TIK[TMK] = KW_EN(TIK, TMK)*	Stored in TSEM non-volatile memory (Flash) for future use when validating the auth factor (KEK[DEK]). It is the start of the EE side key chain.	After use in TSEM volatile memory.	Overwritten with zeroes.
		Read by the TSEM during: • Put_Keys() TSEM command and • Create_Keys() TSEM command.	Used to get the TMK. *TMK = KW_DE(TIK, TIK[TMK])*	After use in TSEM volatile memory.	Overwritten with zeroes.
TMK (TSEM Master Key)	Key Symmetric 256 bits	Created by the TSEM during: • Provision() TSEM command.	Used as a protected key (*TIK[TMK]*) stored in TSEM non-volatile memory (Flash) to be retrieved later for validation (Put_Keys()) and auth factor creation (Create_Keys()). *TIK[TMK] = KW_EN(TIK, TMK)*	After use in TSEM volatile memory.	Overwritten with zeroes.
		Read, wrapped, by the TSEM from non-volatile memory (Flash) during: • Put_Keys() TSEM command and • Create_Keys() TSEM command. And unwrapped by TSEM. *TMK = KW_DE(TIK, TIK[TMK])*	Used to derive the KEK. *KEK = KBKDF(TMK, QEE).*	After use in TSEM volatile memory.	Overwritten with zeroes.

6.6.3 The iECDL Keys and Key Material Lifecycle in Non-Volatile Memory

Table 15: The iECDL Keys and Key Material Lifecycle in Non-Volatile Memory

The Key	Key/KM	When the Key/KM is Introduced into the non-volatile memory.	Use	Where Stored	When the Key/KM is Destroyed	How the Key/KM is destroyed
FW_Sig_Key (FW Signature Key)	Key	The key is created by the Ampex production team when a new TSEM FW build is created. The key is stored in the TSEM non-volatile memory (Flash) at that time.	An ECDSA P-384 with SHA2-384 key stored in TSEM non-volatile memory (Flash). Used by the TSEM at system boot to validate the digital signature of the FW at system boot. It is the public signing key.	Stored protected in the TSEM's non-volatile memory (Flash)[17].	Never destroyed. It is overwritten if a new TSEM FW version is built and burned in TSEM Flash.	It is never destroyed.
KEK[DEK] (The auth factor, the BEV)	KM	Received by the Host System from the TSEM during: • Create_Keys $KEK^{new}[DEK^{new}]$	Received by the Host System from the TSEM during to store the new auth factor in KTD. $KEK^{new}[DEK^{new}]$	KTD's Flash.	The KMs are never destroyed but they may be overwritten.	N/A
PIK (Platform Identity Key	Key	Generated by the TPM manufacturer and stored in non-volatile memory of the Host System (Flash) before delivery of the iECDL.	Read by the Host System and passed to TSEM. Used to derive the TIK (TSEM Identity Key).	Host System non-volatile memory (TPM)	Never erased during the operational life-time of the TOE.	N/A
QEE(Quick Erase Element)	KM	Created by the TSEM and stored in the RSM's non-volatile	Read by the Host System and passed to the TSEM to	RSM non-volatile memory (EEPROM)	Upon invocation of the "Unprovision	Overwritten with zeroes.

[17] Refer to section A.3 for a description of how data stored in a Flash non-volatile memory is protected.

The Key	Key/KM	When the Key/KM is Introduced into the non-volatile memory.	Use	Where Stored	When the Key/KM is Destroyed	How the Key/KM is destroyed
		memory (EEPROM) during: • Provisioning	be used to derive a KEK.		iECDL" GUI Command"[18]	
RSN_SN# (RSN Serial Number) QEE (Quick Erase Element)	KM	Generated by the RSM manufacturer and stored in non-volatile memory of the RSM (EEPROM) before delivery of the iECDL.	Read by the Host System and used to search the KTD for the correct entry to store/or retrieve the new auth factor in the KTD.	RSM non-volatile memory (EEPROM)	Never erased during the operational life-time of the TOE.	N/A
Salt	KM	Created by the TSEM and stored in TSEM's non-volatile memory (Flash) during: • Provisioning.	Read by the TSEM and used to derive the TIK. *TIK = KBKDF(PIK, Salt)*	TSEM non-volatile memory (Flash)	Upon invocation of the "Unprovision IECDL" GUI Command"[19]	Overwritten with zeroes.
KEK[DEK] (KEK wrapped DEK)	KM	Created (wrapped) by the TSEM during: • Create_Keys()	Recieves the new auth factor from the TSEM and stores it on on the KTD. KEK[DEK]	KTD's non-volatile memory (Flash)	Auth Factors are never erased, only overwritten.	N/A

[18] The "Unprovision iECDL" GUI command supports the "Erase DEK" FMT_SMF.1/AA and FMT_SMF.1/EE mandatory cPP management command. Upon invocation, the TSEM calls the destroy() TSEM Command (refer to the KMD).

[19] The "Unprovision iECDL" GUI command supports the "Erase DEK" FMT_SMF.1/AA and FMT_SMF.1/EE mandatory cPP management command. Upon invocation, the TSEM calls the destroy() TSEM Command (refer to the KMD).

7 Terms, Acronyms, Abbreviations, and Definitions

Terms used in this document are defined below.

Table 16: Terms

Acronym	Definition
AA component of the TOE	The portion of the TOE that supports the security functions defined in the FDE_AA cPP. This code is implemented by the ACCE application running on the Host System Card.
auth factor	Authentication Factor
Authentication Factor	A value that a user knows, has, or is (e.g. password, token, etc.) submitted to the TOE to establish that the user is in the community authorized to use the hard disk and that is used in the derivation or decryption of the BEV and eventual decryption of the DEK. Note that these values may or may not be used to establish the particular identity of the user.
Border Encryption Value	A value passed from the AA to the EE intended to link the key chains of the two components.
Clean Shutdown	The process of completely terminating all active processes, services, and applications on a computer system, ensuring that no data is lost or corrupted. Unlike a regular shutdown, which may leave certain processes lingering, a clean shutdown guarantees that all operations are closed properly, which can improve system performance and prevent file system corruption.
Create_Keys() TSEM Command	The Create_Keys() TSEM command creates a new auth factor for a KTD and passes the new auth factor to the Host System. The Host System then overwrites or writes (if the RSM is new) the new key to the KTD.
Cryptographically secure random bytes	Cryptographically secure random bytes are generated using methods that ensure high-entropy and unpredictability, making them suitable for cryptographic application.
CSPRING	Term used to refer to the TSEM Crypto Library's pseudo-random number generator.
Data Encryption Key (DEK)	A key used to encrypt data-at-rest.
Destroy() TSEM Command	Invoked when the Administrator invokes the "Unprovision iECDL" GUI command. The function erases the DEKs, and all key material stored in TSEM Flash (non-volatile memory) and RSM EEPROM (non-volatile memory).
EE component of the TOE	The portion of the TOE that supports the security functions defined in the FDE_EE cPP. This code is implemented by the TSEM Firmware on the TSEM Card.
Erase	Always means overwrite with zeroes in this document.
Graceful System Shutdown	A graceful shutdown is the process of stopping a system, service, or application in a controlled manner, ensuring that ongoing tasks are completed correctly and resources are released appropriately. This method is crucial in distributed systems and microservices to maintain data integrity and avoid disruptions during shutdowns. Unlike a forceful shutdown, which can interrupt running tasks and cause data loss, a graceful shutdown allows all

Acronym	Definition
	process to finish before terminating the service, ensuring user experience and system reliability.
Hot swappable	Hot swappable devices that can be removed or added to a computer system without shutting it down. This technology allows for continuous operation during hardware changes, enhancing usability and efficiency.
Initialize RSM GUI Command	Information held in a TPM is considered secure due to its hardware-based protection. The TPM accts as a secure vault, storing sensitive information like cryptographic keys, passwords, and certificates in an isolated “enclave” that is tamper-resistant. This means that even if the computer is compromised, the information remains protected within the TPM’s secure environment. The TPM’s cryptographic operations ensure that the data is not only stored but also protected from unauthorized access and tampering. Additionally, the TPM’s role in the measured boot process during startup verifies the integrity of the system’s hardware and software, preventing malware from infiltrating the system before it is fully powered on. This combination of hardware-based protection and the ability to validate the system’s integrity during startup makes the TPM a critical component in ensuring the security of sensitive information.
KEK[DEK]	Key DEK is wrapped with key KEK.
Keying Material	Keying material refers to data that is used to establish and maintain cryptographic keying relationships. Keying material is essential for generating symmetric cryptographic keys.
MCXUpresso IDE Launcher	Is an Eclipse-based development environment designed for developers working with NXP’s Arm Cortex-M microcontrollers. (The TSEM’s processor/microarchitecture is NXP Kinetis K81 32-bit Security Microcontroller with microarchitecture ARM Cortex M4). It simplifies the development process by providing a user-friendly interface and a comprehensive suite of tools, including SDKs (Software Development Kit), configuration tools, and debug support. The IDE is optimized for various NXP MCUs and supports features like secure provisioning, configuration tools, and real-time debugging, making it an essential tool for embedded system development.
Netcat (nc)	The nc command is a versatile networking utility in RHEL used for reading and writing data across network connections using TCP or UDP protocols.
Non-volatile memory	Keys are considered protected when stored in non-volatile memory, which ensures that the data remains secure even when the device is powered off.
Not zeroed in RAM when shutdown	When a computer is shut down, the contents of RAM are cleared, and any sensitive data, such as encryption keys, are no longer stored in the memory. This is because RAM is volatile memory type, meaning it requires a constant power supply to retain data. Once power is turned off, the RAM loses its contents and is reset, ready to be filed with new data when the computer restarts. Therefore, keys are not zeroed in RAM when shutdown, but they are not stored in RAM either.
Provision() TSEM Command	Creates the key material stored in the TSEM non-volatile memory (Flash) and the RSM’s non-volatile memory (EEPROM). Specifically, create the Salt, derive the TIK from the PIK, creates the TMK, wraps the TMK with the TIK, and stores

Acronym	Definition
	the TIK[TMK] in TSEM Flash for use of the beginning of the EE side key chain. Also, the function creates the QEE and stores it in RSM EEPROM.
Put_Keys() TSEM Command	Load the SATA Controllers with the DEKs. The system must be provisioned. Specifically, derive and decrypt keys in the key chain utile the KEK is derived. Unwrap the received auth factor KEK[DEK] with the derived KEK. If an error occurs, validation fails, and the DEKs are not loaded. Otherwise, the DEKs are loaded.
"Renitialize RSM" GUI Command	
Root User	The **root** user is the default highest privileged account in Unix-like systems. It has a user identifier (UID) of 0 and can perform any operation on the system, including creating/deleting user accounts, modifying system files, and running any command. The root account is typically locked or recommended not to be used for regular tasks to prevent accidental system damage.
Secure Erase	Secure Erase is the data sanitization method that complexly wipes out all data on a storage device to prevent any possibility of data recovery using industry-standard techniques. It works by overwriting the entire storage area multiple times with random or predefined patterns, making the original data unreadable. Secure Erase is a vital process in maintaining data confidentiality and preventing data breaches when disposing of or repurposing storage devices.
Submasks	*[FDE_AA] Line 7&8, definition of a submask: "The TOE may accept any number of authorization factors, and these are categorized as 'submasks'".*
Sudo and Superuser	The sudo command allows a permitted user to execute a command as the superuser or another user, as specified by the security policy. By default, sudo runs commands as the root user, but it can be configured to run commands as other users. This means that using sudo does not always mean running commands as the root user, but it often does in practice.
Superuser	The term **superuser** is a generic term that refers to any user account with elevated privileges, similar to the root user. In most Unix-like systems, the root user is the primary superuser. However, other user accounts can also be granted superuser privileges by adding them to a special group, such as the wheel group. These accounts can perform system administration tasks but are not necessarily the root user.
Sync	The iECDL Sync command copies the data of a KTD to a cRSM.
TIK[TMK]	The TMK (TSEM Master Key) wrapped with the TIK (TSEM Identity Key)
TPM Hardware-based Protection	Information held in a TPM is considered secure due to its hardware-based protection. The TPM accts as a secure vault, storing sensitive information like cryptographic keys, passwords, and certificates in an isolated "enclave" that is tamper-resistant. This means that even if the computer is compromised, the information remains protected within the TPM's secure environment. The TPM's cryptographic operations ensure that the data is not only stored but also protected from unauthorized access and tampering. Additionally, the TPM's role in the measured boot process during startup verifies the integrity of the system's hardware and software, preventing malware from infiltrating the system before it is fully powered on. This combination of hardware-based protection and the ability to validate the system's integrity during startup

Acronym	Definition
	makes the TPM a critical component in ensuring the security of sensitive information.
TRIM	A command used that allows the operating system to inform the drive which data blocks are no longer in use, enabling the drive to erase them and improve performance.
TSEM Library	A library, included in the ACCE application, that enables the Host System to communicate with the TSEM. It includes four calls provision, create keys, put keys, and destroy.
TSF Data	Data included in the TOE's physical boundary.
"Unprovision iECDL" GUI Command	The "Unprovision iECDL" GUI command supports the "Erase DEK" FMT_SMF.1/AA and FMT_SMF.1/EE mandatory management command. Upon invocation, the TSEM calls the destroy() TSEM function (refer to the KMD).

Acronyms and Abbreviations used in this document are defined in the following table.

Table 17: Acronyms and Abbreviations

Acronym	Definition
AA	Authorization Acquisition
AC	Alternating Current
ACCE	Ampex Common Computing Environment
ACPI	Advanced Configuration and Power Interface
AES	Advanced Encryption Standard
AGD	Administrative Guidance
BEV	Border Encryption Value
CAST	Cryptographic Algorithm Self Tests
CAVP	Cryptographic Algorithm Verification Program
CC	Common Criteria
CLI	Command Line Interface
cPP	Collaborative Protection Profile
CPU	Central Processing Unit
cRSM	compact Removable Storage Media
CSfC	Commercial Solutions for Classified
DRBG	Deterministic Random Bit Generator
DEK	Data Encryption Key
ECDSA	Elliptic Curve Digital Signature Algorithm
EE	Encryption Engine
EEPROM	Electrically Erasable Programmable Random Access Memory
EP	Extended Package
FDE	Full Drive Encryption

Acronym	Definition
FDE_AA	*collaborative Full Drive Encryption – Authorization Acquisition Protection Profile*
FDE_EE	*collaborative Full Drive Encryption – Encryption Engine Protection Profile*
FPGA	Field Programmable Gate Array
FQDN	Fully Qualified Domain Name
GB	Giga Byte
Gb	Gigabit
GbE	Gigabit Ethernet
GNU	GNU's Not Unix.
GPG	GNU Privacy Guard
GUI	Graphical User Interface
HMAC	Keyed-Hash Message Authentication Code
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IDE	Integrated Development Environment
iECDL	Intelligent E-2 Common Data Loader
iRSM	Intelligent Removable Storage Media
IP	Internet Protocol
IV	Initialization Vector
KBKDF	Key-Based Key Derivation Function
KEK	Key Encryption Key
KTD	Key Transfer Device
KW	Key Wrap
LAN	Local Area Network
LED	Light Emitting Diode
MCU	Microcontroller Unit
N/A	Not Applicable
NAND	Negative-AND (gate)
Nc	Netcat
NIAP	Nation Information Assurance Partnership
NFS	Network File System
NSR	Not Security Relevant
NVM	Non-Volatile Memory
OE	Operational Environment
OS	Operating System
OSP	Organizational Security Policy
PBKDF2	Password-based Key Derivation Function 2
PC	Personal Computer

Acronym	Definition
PCL	Product Compliant List
PIK	Platform Identity Key
PP	Protection Profile
QEE	Quick Erase Element
RAP	Register Access Protection
RASP	Removable Advanced Storage Pack
RBG	Random Bit Generator
RHEL	Red Hat Enterprise Linux
RPM	RPM Package Manager (a recursive algorithm)
RSM	Removable Storage Media
SATA	Serial Advanced Technology Attachment
SCP	Secure Copy Protocol
SDK	Software Development Kit
SE	Secure Element
SFR	Security Functional Requirement
SoC	System on Chip
SSD	Solid State Drive
SSH	Secure Shell
ST	Security Target
TD	Technical Decision
TCP	Transmission Control Protocol
TFTP	Trivial File Transfer Protocol
TIK	TSEM Identity Key
TLS	Transport Layer Security
TOE	Target of Evaluation
TPM	Trusted Platform Module
TSEM	TuffServ® Encryption Module
TOE	Target of Evaluation
TSS	TOE Summary Specification
USB	Universal Serial Bus
V	Voltage
VMM	Virtual Memory Manager
XTS	XEX (XOR Encrypt XOR) Tweakable Block Cipher with Ciphertext Stealing

8 Security Target Conventions

8.1 Conventions

The following sections identify and describe the syntax used throughout the document to depict keys and Security Elements (SEs).

8.1.1 Colors

Keys are depicted in red font.

Security Elements (SEs) are depicted in green font.

8.1.2 Key Based Key Derivation Function (KBKDF)

The TOE uses a Key Based Key Derivation Function (KBKDF) to create a derived key. Throughout this document the notation used to depict this function is described below.

Derived Key (also referred to as Key Material) = KBKDF(Key Derivation Key, Context)

8.1.3 Key Wrap/Unwrap Functions

The TOE wraps keys and unwraps keys. Throughout this document the notation used to depict these functions is described below.

Key Wrap Function

KeyA[KeyB] = KW_EN(KeyA, KeyB)

The key wrap function KW_EN is called to wrap KeyB with KeyA and return the wrapped key.

Key Unwrap Function

KeyB = KW_DE(KeyA, KeyA[KeyB])

A wrapped key, KeyA[KeyB], is thought to be wrapped with KeyA.
The command KW_DE() unwraps the key and returns the unwrapped key.
If the wrapped key was not wrapped with KeyA, an error will be returned.
Otherwise, the unwrap was successful, and KeyB is returned.

8.1.4 Key Distinction

In this document the same value is being compared. When that happens, the value is appended with a string in the exponent field identifying the source e.g. Key Encryption Key, KEK^{TSEM} means the KEK is from the TSEM (TuffServ® Security and Encryption Module) verses KEK^{KTD} means the KEK is from the KTD (Key Transfer Device).

When new keys and SEs are replacing the current keys and SEs, the value is appended with a string in the exponent field indicating new e.g. TSEM Identify Key, TIK^{new} means the TIK is new verses the current TIK designated as TIK.

Appendix A

A.1 Differences Between the RSM and iRSM

The differences between RSM and iRSM.

1. **Basic Functionality:**
    - **RSM:** Primarily used for storing and transferring data between devices. It includes simple storage solutions like USB flash drives, CDs, DVDs, and external hard drives.
    - **iRSM:** Includes all basic functionalities of RSM but adds advanced features to enhance performance and security.
2. **Security Features:**
    - **RSM:** Generally, basic models have minimal security features, although some may offer password protection.
    - **iRSM:** Typically includes advanced security measures such as hardware-based encryption, biometric access controls, and secure data erasure to prevent unauthorized access.
3. **Data Management:**
    - **RSM:** Basic data storage without any built-in management capabilities.
    - **iRSM:** Often features intelligent data management like automated backup, data compression, and organization tools to improve efficiency.
4. **Performance Optimization:**
    - **RSM:** Standard performance based on the hardware specifications.
    - **iRSM:** Enhanced performance with features like faster data transfer rates, improved read/write speeds, and optimization algorithms to reduce latency.
5. **Integration with Systems:**
    - **RSM:** Simple plug-and-play devices compatible with most systems without additional software.
    - **iRSM:** Often includes specialized software or firmware that integrates with the host system for better control and monitoring of the device.

In essence, iRSM offers a smarter, more secure, and efficient approach to data storage compared to traditional RSM.

A.2 Compliant-power Saving State Definitions

Key
Green text indicates states supported by the two FDE cPPs.
Green underlined text indicates the states the ST claims.

Table 18: System Power State Definitions

Power State	ACPI state	Description
Working	S0	The system is fully usable. Hardware components that aren't in use can save power by entering a lower power state.
Sleep (Modern Standby)	*S0* low-power idle	Some SoC systems support a low-power idle state known as Modern Standby. In this state, the system can very quickly switch from a low-power state to high-power state in response to hardware and network events. **Note:** SoC systems that support Modern Standby don't use *S1-S3*.
Sleep	S1 S2 S3	The system appears to be off. The amount of power consumed in states *S1-S3* is less than *S0* and more than *S4*. *S3* consumes less power than *S2*, and *S2* consumes less power than *S1*. Systems typically support one of these three states, not all three. In states *S1-S3*, volatile memory is kept refreshed to maintain the system state. Some components remain powered so the computer can wake from input from the keyboard, LAN, or a USB device. *Hybrid sleep*, used on desktops, is where a system uses a hibernation file with *S1-S3*. The hibernation file saves the system state in case the system loses power while in sleep. **Note:** SoC systems that support Modern Standby don't use *S1-S3*.
Hibernate	S4	The system appears to be off. Power consumption is reduced to the lowest level. The system saves the contents of volatile memory to a hibernation file to preserve system state. Some components remain powered so the computer can wake from input from the keyboard, LAN, or a USB device. The working context can be restored if it's stored on nonvolatile media. *Fast startup* is where the user is logged off before the hibernation file is created. This allows for a smaller hibernation file, more appropriate for systems with less storage capabilities.
Soft off	S5	The system appears to be off. This state is comprised of a full shutdown and boot cycle.
	G2	The system is in a low-power mode, allowing for energy savings while retaining enough context to quickly resume operations upon input from the keyboard, LAN, or USB devices.
	G2(S5)	In this state the operating system has shut down, and the system context is not retained, requiring a complete reboot to return to the working state.
Mechanical off	G3	The system is completely off and consumes no power. The system returns to the working state only after a full reboot.

Appendix B

The following tables are included as an Appendix to illustrate the two cPPs SFR claims are complete individually and can stand alone.

Key:
M = Mandatory SFR
S = Selection-Based SFR
O = Optional Requirement

Table 19: FDE_AA cPP SFRs

Requirement	M /S	SFR That Prompts the Inclusion of the Selection-Based SFR or Application Note
Cryptographic Support (FCS)		
FCS_AFA_EXT.1 Authorization Factor Acquisition *(FDE_AA)*	M	N/A
FCS_AFA_EXT.2 Timing of Authorization Factor Acquisition *(FDE_AA)*	M	N/A
FCS_CKM.4(a)/AA Cryptographic Key Destruction (Power Management) *(FDE_AA)*	M	N/A
FCS_CKM.4(d) Cryptographic Key Destruction (Software TOE, 3rd Party Storage) *(FDE_AA)*	M	N/A
FCS_CKM_EXT.4(a) Cryptographic Key and Key Material Destruction (Destruction Timing) *(FDE_AA) (FDE_EE)*	M	N/A
FCS_CKM_EXT.4(b)/AA Cryptographic Key and Key Material Destruction (Power Management) *(FDE_AA)*	M	N/A
FCS_COP.1(a)/AA Cryptographic Operation (Signature Verification) *(FDE_AA)*	S	FPT_TUD_EXT.1/AA Trusted Update *(FDE_AA)*
FCS_COP.1(b)/AA Cryptographic Operation (Hash Algorithm) *(FDE_AA)*	S	FCS_COP.1(a)/AA Cryptographic Operation (Signature Verification) *(FDE_AA)*
FCS_KYC_EXT.1 Key Chaining (Initiator) *(FDE_AA)*	M	N/A
FCS_SNI_EXT.1/AA Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) *(FDE_AA)*	M	N/A
Security Management (FMT)		
FMT_MOF.1 Management of Functions Behavior *(FDE_AA)*	M	N/A
FMT_SMF.1/AA Specification Management Functions *(FDE_AA)*	M	N/A
FMT_SMR.1 Security Roles *(FDE_AA)*	M	N/A
Protection of the TSF (FPT)		
FPT_KYP_EXT.1/AA Protection of Key and Key Material *(FDE_AA)*	M	N/A

Requirement	M /S	SFR That Prompts the Inclusion of the Selection-Based SFR or Application Note
FPT_PWR_EXT.1/AA Power Savings State *(FDE_AA)*	M	N/A
FPT_PWR_EXT.2/AA Timing of Power Savings States *(FDE_AA)*	M	N/A
FPT_TST_EXT.1/AA TSF Testing *(FDE_AA)*	O	N/A
FPT_TUD_EXT.1/AA Trusted Update *(FDE_AA)*	M	N/A

Table 20: FDE EE cPP SFRs

Requirement	M /S	SFR That Prompts the Inclusion of the Selection-Based SFR or Application Note
Cryptographic Support (FCS)		
FCS_CKM.1(b) Cryptographic Key Generation (Symmetric keys) *(FDE_EE)*	S	FCS_KYC_EXT.2 Key Chaining (Recipient) *(FDE_EE)*
FCS_CKM.1(c) Cryptographic Key Generation (Data Encryption Key) *(FDE_EE)*	M	N/A
FCS_CKM.4(a)/EE Cryptographic Key Destruction (Power Management) *(FDE_EE)*	M	N/A
FCS_CKM.4(b) Cryptographic Key Destruction (TOE-Controlled Hardware) *(FDE_EE)*	S	FCS_CKM_EXT.6 Cryptographic Key Destruction Types *(FDE_EE)*
FCS_CKM_EXT.4(a) Cryptographic Key and Key Material Destruction (Destruction Timing) *(FDE_AA) (FDE_EE)*	M	N/A
FCS_CKM_EXT.4(b)/EE Cryptographic Key and Key Material Destruction (Power Management) *(FDE_EE)*	M	N/A
FCS_CKM_EXT.6 Cryptographic Key Destruction Types *(FDE_EE)*	M	N/A
FCS_COP.1(a)/EE Cryptographic Operation (Signature Verification) *(FDE_EE)*	S	FPT_TUD_EXT.1/EE Trusted Update *(FDE_EE)*
FCS_COP.1(b)/EE Cryptographic Operation (Hash Algorithm) *(FDE_EE)*	S	FCS_KDF_EXT.1 Cryptographic Key Derivation *(FDE_EE)*
FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm) *(FDE_EE)*	S	FCS_KDF_EXT.1 Cryptographic Key Derivation *(FDE_EE)*
FCS_COP.1(d) Cryptographic Operation (Key Wrapping) *(FDE_EE)*	S	FCS_CKM.1(c) Cryptographic Key Generation (Data Encryption Key) *(FDE_EE)* FCS_KYC_EXT.2 Key Chaining (Recipient) *(FDE_EE)* FCS_VAL_EXT.1 Validation *(FDE_EE)* FPT_KYP_EXT.1/EE Protection of Key and Key Material *(FDE_EE)*

Requirement	M /S	SFR That Prompts the Inclusion of the Selection-Based SFR or Application Note
FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption) *(FDE_EE)*	S	FDP_DSK_EXT.1 Protection of Data on Disk *(FDE_EE)*
FCS_KDF_EXT.1 Cryptographic Key Derivation *(FDE_EE)*	S	FCS_KYC_EXT.2 Key Chaining (Recipient) *(FDE_EE)*
FCS_KYC_EXT.1 Key Chaining (Recipient) *(FDE_EE)*	M	N/A
FCS_RBG_EXT.1 Random Bit Generation *(FDE_EE)* ,	S	FCS_CKM.1(b) Cryptographic Key Generation (Symmetric Keys) *(FDE_EE)*
FCS_SNI_EXT.1/EE Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) *(FDE_EE)*	M	N/A
FCS_VAL_EXT.1 Validation *(FDE_EE)*	M	N/A
User Data Protection (FDP)		
FDP_DSK_EXT.1 Protection of Data on Disk *(FDE_EE)*	M	N/A
Security Management (FMT)		
FMT_SMF.1/EE Specification Management Functions *(FDE_EE)*	M	N/A
Protection of the TSF (FPT)		
FPT_FUA_EXT.1 Firmware Update Authentication *(FDE_EE)*	S	FPT_TUD_EXT.1/EE
FPT_KYP_EXT.1/EE Protection of Key and Key Material *(FDE_EE)*	M	N/A
FPT_PWR_EXT.1/EE Power Savings State *(FDE_EE)*	M	N/A
FPT_PWR_EXT.2/EE Timing of Power Savings States *(FDE_EE)*	M	N/A
FPT_TST_EXT.1/EE TSF Testing *(FDE_EE)*	M	N/A
FPT_TUD_EXT.1/EE Trusted Update *(FDE_EE)*	M	N/A