Cigent PBA Software with Cigent M.2 2230 PCIe Gen 4 Self-Encrypting Drive (SED) Security Target

Version 1.3
11/11/25

Prepared for:

Cigent Technology, Inc.

2211 Widman Way, Suite 150
Fort Myers, Florida 33901

Prepared By:

Gossamer Laboratories

www.gossamersec.com

LIST OF TABLES

1. Security Target Introduction

This section identifies the Security Target (ST) and Target of Evaluation (TOE) identification, ST conventions, ST conformance claims, and the ST organization. The TOE is Cigent PBA Software with Cigent M.2 2230 PCIe Gen 4 Self-Encrypting Drive (SED) provided by Cigent Technology, Inc. The TOE is being evaluated as a full drive encryption solution.

The Security Target contains the following additional sections:

- Conformance Claims (Section 2)
- Security Objectives (Section 3)
- Extended Components Definition (Section 4)
- Security Requirements (Section 5)
- TOE Summary Specification (Section 6)

ConventionsH

The following conventions have been applied in this document:

- Security Functional Requirements – Part 2 of the CC defines the approved set of operations that may be applied to functional requirements: iteration, assignment, selection, and refinement.
  - Iteration: allows a component to be used more than once with varying operations. In this ST, iteration may be indicated by a parenthetical number placed at the end of the component. For example FDP_ACC.1(1) and FDP_ACC.1(2) indicate that the ST includes two iterations of the FDP_ACC.1 requirement. Alternately, a usually descriptive textual extension may be added after a slash (/) character to identify a specific iteration. For example, iterations of a requirement such as FCS_COP.1 might be identified as FCS_COP.1/HASH and FCS_COP.1/CRYPT.
  - Assignment: allows the specification of an identified parameter. Assignments are indicated using bold and are surrounded by brackets (e.g., [**assignment**]). Note that an assignment within a selection would be identified in italics and with embedded bold brackets (e.g., [***[selected-assignment]***]).
  - Selection: allows the specification of one or more elements from a list. Selections are indicated using bold italics and are surrounded by brackets (e.g., [***selection***]).
  - Refinement: allows the addition of details. Refinements are indicated using bold, for additions, and strike-through, for deletions (e.g., "… **all** objects …" or "… ~~some~~ **big** things …").
- Other sections of the ST – Other sections of the ST use bolding to highlight text of special interest, such as captions.

1.1 Security Target Reference

ST Title – Cigent PBA Software with Cigent M.2 2230 PCIe Gen 4 Self-Encrypting Drive (SED) Security Target

ST Version – Version 1.3

ST Date – 11/11/25

1.2 TOE Reference

TOE Identification – Cigent Technology, Inc. Cigent PBA Software version 2.0 with Cigent M.2 2230 PCIe Gen 4 Self-Encrypting Drive (SED)

TOE Developer – Cigent Technology, Inc.

Evaluation Sponsor – Cigent Technology, Inc.

1.3 TOE Overview

The Target of Evaluation (TOE) is the Cigent PBA Software version 2.0 with Cigent M.2 2230 PCIe Gen 4 Self-Encrypting Drive (SED). Cigent offers the M.2 2230 PCIe Gen 4 in several different capacities (128GB-1TB) and two different form factors (BGA or M.2). No matter the capacity or form factor, the Cigent SED operates in the same manner, providing the same encryption, key management, and security. Irrespective of whether the SED comes in the Ball Grid Array (BGA) form factor, meant to be soldered to a computing system's motherboard or whether the SED has an M.2 form factor (which allows a user to remove or upgrade an M.2 SED within their laptop), the below SEDs all provide identical security features and provide hardware full drive encryption.

Controller	Firmware	Capacity	Form Factor	Model Name
PS5021-E21T	ELFCI0C.1	64GB	M.2	CGN-0C3014_064G
PS5021-E21T	ELFCI0C.1	128GB	M.2	CGN-0C3014_128G
PS5021-E21T	ELFCI0C.1	256GB	M.2	CGN-0C3014_256G
PS5021-E21T	ELFCI0C.1	512GB	M.2	CGN-0C3014_512G
PS5021-E21T	ELFCI0C.1	1TB	M.2	CGN-0C3014_001T
PS5021-E21T	ELFCI0C.1	64GB	M.2	CGN-0C1014_064G
PS5021-E21T	ELFCI0C.1	128GB	M.2	CGN-0C1014_128G
PS5021-E21T	ELFCI0C.1	256GB	M.2	CGN-0C1014_256G
PS5021-E21T	ELFCI0C.1	512GB	M.2	CGN-0C1014_512G
PS5021-E21T	ELFCI0C.1	1TB	M.2	CGN-0C1014_001T
PS5021-E21T	ELFCI0C.1	64GB	M.2	CGN-0C2014_064G
PS5021-E21T	ELFCI0C.1	64GB	M.2	CGN-0C2014_064G
PS5021-E21T	ELFCI0C.1	128GB	M.2	CGN-0C2014_128G
PS5021-E21T	ELFCI0C.1	256GB	M.2	CGN-0C2014_256G
PS5021-E21T	ELFCI0C.1	512GB	M.2	CGN-0C2014_512G
PS5021-E21T	ELFCI0C.1	1TB	M.2	CGN-0C2014_001T
PS5021-E21T	ELFCI0C.1	64GB	M.2	CGN-0C2014_064G
PS5021-E21T	ELFCI0C.1	128GB	M.2	CGN-9C3014_128G
PS5021-E21T	ELFCI0C.1	256GB	M.2	CGN-9C3014_256G
PS5021-E21T	ELFCI0C.1	512GB	M.2	CGN-9C3014_512G
PS5021-E21T	ELFCI0C.1	1TB	M.2	CGN-9C3014_001T
PS5021-E21T	ELFCI0C.1	64GB	BGA	CGN-0C3304_064G
PS5021-E21T	ELFCI0C.1	128GB	BGA	CGN-0C3304_128G
PS5021-E21T	ELFCI0C.1	256GB	BGA	CGN-0C3304_256G
PS5021-E21T	ELFCI0C.1	512GB	BGA	CGN-0C3304_512G
PS5021-E21T	ELFCI0C.1	1TB	BGA	CGN-0C3304_001T

1.4 TOE Description

The TOE consists of software that provides pre-boot authentication (PBA) and a hardware M.2 2230 PCIe Gen 4 self-encrypting drive (SED).

The TOE's Pre-Boot user Authentication (PBA) for Opal 2.0 compliant SEDs supporting MBR Shadowing. . It has been tested on a Dell Precision 3591 laptop with an Intel Core Ultra 7 155H (Series 1/Meteor Lake) CPU with the Cigent M.2 2230 PCIe Gen 4 SED as a coupled system to deliver secure Data-At-Rest (DAR) encryption.

The TOE's PBA software is installed on a 128MB read-only Shadow partition on the M.2 2230 PCIe Gen 4 SED. After installation, the PBA allows the user to authenticate, which unlocks the SED and boots to the protected OS environment.

The Cigent M.2 2230 PCIe Gen 4 SED provides encrypted storage to protect data until the SED has successfully received the Border Encryption Validation (BEV) from an Authorization Acquisition component (like Cigent's PBA). Cigent's M.2 2230 PCIe Gen 4, by nature of its Opal 2.0 compliance with support for MBR Shadowing, can interoperate with any pre-boot authentication software designed to interact with Opal 2.0 SEDs supporting MBR Shadowing and FDE AA certified.

The Cigent PBA Software acts as an Authorization Acquisition component that supplies the Border Encryption Validation (BEV) to an Encryption Engine component (like Cigent's M.2 2230 PCIe Gen 4). The PBA software can interoperate with any FDE EE certified SSD supporting Opal 2.0 and MBR shadowing.

1.4.1 TOE Architecture

The TOE is composed of two components. The first, Cigent's PBA software, provides Authorization Acquisition software that runs on any computer with an x86 compatible CPU. The PBA presents a user with a graphic interface to enter the password and/or access a physical token (smartcard, FIDO2 security key, or USB drive) or a combination of password and a physical token. The second component, Cigent's M2. 2230 Self Encryption Drive (SED), provides the Encryption Engine as well as solid state storage to securely house the user's entire operating system and data files. The SED only allows presents drive access to the separate Shadow partition area (that stores the PBA software) until the SED receives a valid secret value (the BEV). The PBA provides this BEV value after deriving the BEV from operator provided authorization factor(s). After validating the BEV, the SED uses the BEV to derive keys to cryptographically unlock the drive, allowing the user access (enabling the user to boot to the protected OS as well as allowing the user to access data files and applications).

1.4.1.1 Physical Boundaries

The TOE's PBA component is purely software that one installs onto the drive within an x86-based computer. The TOE's SED component is a secure, solid state storage device in an M.2 or BGA form factor.

1.4.1.2 Logical Boundaries

This section summarizes the security functions provided by the Cigent PBA with Cigent M.2 2230 PCIe Gen 4:

- Cryptographic support
- User data protection
- Security management
- Protection of the TSF

1.4.1.2.1 Cryptographic support

The TOE includes cryptographic functionality for key management, user authentication, and block-based encryption including: symmetric key generation, encryption/decryption, cryptographic hashing, keyed-hash message authentication, and password-based key derivation. These functions are supported with suitable random bit generation, key derivation, salt generation, initialization vector generation, secure key storage, and key destruction. These primitive cryptographic functions are used to encrypt Data-At-Rest (including the generation and protection of keys and key encryption keys) used by the TOE.

1.4.1.2.2 User data protection

The TOE performs Full Drive Encryption on all visible (the SED maintains the Shadow partition in plaintext prior to authentication) partitions on the drive (so that no plaintext exists) and does so without user intervention.

1.4.1.2.3 Security management

The TOE provides each of required management services to manage the full drive encryption using a graphical user interface.

1.4.1.2.4 Protection of the TSF

The TOE implements a number of features to protect itself to ensure the reliability and integrity of its security features. It protects key and key material, and includes functions to perform self-tests and software/firmware integrity checking so that it might detect when it is failing or may be corrupt. If any of the self-tests fail, the TOE will not go into an operational mode.

1.4.2 TOE Documentation

Cigent Single and Multidrive PBA Installation Guide and User Manual June 2025, PBA Version 2.0.0 [Admin Guide]

2. Conformance Claims

This TOE is conformant to the following CC specifications:

- Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, Version 3.1, Revision 5, April 2017.
    - Part 2 Extended
- Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, Version 3.1, Revision 5, April 2017.
    - Part 3 Conformant
- Package Claims:
    - collaborative Protection Profile for Full Drive Encryption - Authorization Acquisition, Version 2.0 + Errata 20190201, 01 February 2019(FDEAAcPP20E)
    - collaborative Protection Profile for Full Drive Encryption - Encryption Engine, Version 2.0 + Errata 20190201, 01 February 2019 (FDEEEcPP20E)

Package	Technical Decision	Applied	Notes
CPP_FDE_AA_V2.0E	TD0458 - FIT Technical Decision for FPT_KYP_EXT.1 evaluation activities	Yes	
CPP_FDE_AA_V2.0E	TD0606 - FIT Technical Recommendation for Evaluating a NAS against the FDE AA and FDEE	No	Product not a NAS
CPP_FDE_AA_V2.0E	TD0759 - FIT Technical Decision for FCS_AFA_EXT.1.1	Yes	
CPP_FDE_AA_V2.0E	TD0760 - FIT Technical Decision for FCS_SNI_EXT.1.3, FCS_COP.1(f)	Yes	
CPP_FDE_AA_V2.0E	TD0765 - FIT Technical Decision for FMT_MOF.1	Yes	
CPP_FDE_AA_V2.0E	TD0766 - FIT Technical Decision for FCS_CKM.4(d) Test Notes	Yes	
CPP_FDE_AA_V2.0E	TD0767 - FIT Technical Decision for FMT_SMF.1.1	Yes	
CPP_FDE_AA_V2.0E	TD0769 - FIT Technical Decision for FPT_KYP_EXT.1.1	No	
CPP_FDE_AA_V2.0E	TD0929 - FIT Technical Decision: Clarification to FCS_PCC_EXT.1.1	Yes	
CPP_FDE_EE_V2.0E	TD0458 - FIT Technical Decision for FPT_KYP_EXT.1 evaluation activities	Yes	
CPP_FDE_EE_V2.0E	TD0460 - FIT Technical Decision for FPT_PWR_EXT.1 non-compliant power saving states	Yes	
CPP_FDE_EE_V2.0E	TD0464 - FIT Technical Decision for FPT_PWR_EXT.1 compliant power saving states	Yes	
CPP_FDE_EE_V2.0E	TD0606 - FIT Technical Recommendation for Evaluating a NAS against the FDE AA and FDEE	No	Product not a NAS
CPP_FDE_EE_V2.0E	TD0766 - FIT Technical Decision for FCS_CKM.4(d) Test Notes	No	SFR not selected
CPP_FDE_EE_V2.0E	TD0769 - FIT Technical Decision for FPT_KYP_EXT.1.1	No	

Table 1 Technical Decisions

2.1 Conformance Rationale

The ST conforms to the FDEAAcPP20E/FDEEEcPP20E. The security problem definition, security objectives, and security requirements are referenced and available in the PP(s) listed above.

3. Security Objectives

The Security Problem Definition may be found in the FDEAAcPP20E/FDEEEcPP20E and this section reproduces only the corresponding Security Objectives for operational environment for reader convenience. The FDEAAcPP20E/FDEEEcPP20E offers additional information about the identified security objectives, but that has not been reproduced here and the FDEAAcPP20E/FDEEEcPP20E should be consulted if there is interest in that material.

In general, the FDEAAcPP20E/FDEEEcPP20E has defined Security Objectives appropriate for full drive encryption solution and as such are applicable to the Cigent PBA Software with Cigent M.2 2230 PCIe Gen 4 Self-Encrypting Drive (SED) TOE.

3.1 Security Objectives for the Operational Environment

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 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 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 Volatile memory is cleared after power-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 and EE) is sufficiently protected to prevent information disclosure.

4. Extended Components Definition

All of the extended requirements in this ST have been drawn from the FDEAAcPP20E/FDEEEcPP20E. The FDEAAcPP20E/FDEEEcPP20E defines the following extended requirements and since they are not redefined in this ST the FDEAAcPP20E/FDEEEcPP20E should be consulted for more information in regard to those CC extensions.

Extended SFRs:

- FDEAAcPP20E:FCS_AFA_EXT.1: Authorization Factor Acquisition - per TD0759
- FDEAAcPP20E:FCS_AFA_EXT.2: Timing of Authorization Factor Acquisition
- FDEAAcPP20E:FCS_CKM_EXT.4(a): Cryptographic Key and Key Material Destruction (Destruction Timing)
- FDEEEcPP20E:FCS_CKM_EXT.4(a): Cryptographic Key and Key Material Destruction (Destruction Timing)
- FDEAAcPP20E:FCS_CKM_EXT.4(b): Cryptographic Key and Key Material Destruction (Power Management)
- FDEEEcPP20E:FCS_CKM_EXT.4(b): Cryptographic Key and Key Material Destruction (Power Management)
- FDEEEcPP20E:FCS_CKM_EXT.6: Cryptographic Key Destruction Types
- FDEAAcPP20E:FCS_KDF_EXT.1: Cryptographic Key Derivation
- FDEAAcPP20E:FCS_KYC_EXT.1: Key Chaining (Initiator)
- FDEEEcPP20E:FCS_KYC_EXT.2: Key Chaining (Recipient)
- FDEAAcPP20E:FCS_PCC_EXT.1: Cryptographic Password Construct and Conditioning
- FDEAAcPP20E:FCS_RBG_EXT.1: Extended: Cryptographic Operation (Random Bit Generation)
- FDEEEcPP20E:FCS_RBG_EXT.1: Random Bit Generation
- FDEAAcPP20E:FCS_SMC_EXT.1: Submask Combining
- FDEAAcPP20E:FCS_SNI_EXT.1: Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) - per TD0760
- FDEEEcPP20E:FCS_SNI_EXT.1: Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation)
- FDEEEcPP20E:FCS_VAL_EXT.1: Validation
- FDEEEcPP20E:FDP_DSK_EXT.1: Protection of Data on Disk
- FDEEEcPP20E:FPT_FUA_EXT.1: Firmware Update Authentication
- FDEAAcPP20E:FPT_KYP_EXT.1: Protection of Key and Key Material
- FDEEEcPP20E:FPT_KYP_EXT.1: Protection of Key and Key Material
- FDEAAcPP20E:FPT_PWR_EXT.1: Power Saving States
- FDEEEcPP20E:FPT_PWR_EXT.1: Power Saving States
- FDEAAcPP20E:FPT_PWR_EXT.2: Timing of Power Saving States
- FDEEEcPP20E:FPT_PWR_EXT.2: Timing of Power Saving States
- FDEEEcPP20E:FPT_RBP_EXT.1: Rollback Protection
- FDEAAcPP20E:FPT_TST_EXT.1: TSF Testing
- FDEEEcPP20E:FPT_TST_EXT.1: TSF Testing
- FDEAAcPP20E:FPT_TUD_EXT.1: Trusted Update
- FDEEEcPP20E:FPT_TUD_EXT.1: Trusted Update

5. Security Requirements

This section defines the Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs) that serve to represent the security functional claims for the Target of Evaluation (TOE) and to scope the evaluation effort.

The SFRs have all been drawn from the FDEAAcPP20E/FDEEEcPP20E. The refinements and operations already performed in the FDEAAcPP20E/FDEEEcPP20E are not identified (e.g., highlighted) here, rather the requirements have been copied from the FDEAAcPP20E/FDEEEcPP20E and any residual operations have been completed herein. Of particular note, the FDEAAcPP20E/FDEEEcPP20E made a number of refinements and completed some of the SFR operations defined in the Common Criteria (CC) and that PP should be consulted to identify those changes if necessary.

The SARs are also drawn from the FDEAAcPP20E/FDEEEcPP20E. The FDEAAcPP20E/FDEEEcPP20E should be consulted for the assurance activity definitions.

5.1 TOE Security Functional Requirements

The following table identifies the SFRs that are satisfied by Cigent PBA Software with Cigent M.2 2230 PCIe Gen 4 Self-Encrypting Drive (SED) TOE.

Requirement Class	Requirement Component
FCS: Cryptographic support	FDEAAcPP20E:FCS_AFA_EXT.1: Authorization Factor Acquisition - per TD0759
	FDEAAcPP20E:FCS_AFA_EXT.2: Timing of Authorization Factor Acquisition
	FDEAAcPP20E:FCS_CKM.1(b): Cryptographic key generation (Symmetric Keys)
	FDEEEcPP20E:FCS_CKM.1(c): Cryptographic Key Generation (Data Encryption Key)
	FDEAAcPP20E:FCS_CKM.4(a): Cryptographic Key Destruction (Power Management)
	FDEEEcPP20E:FCS_CKM.4(a): Cryptographic Key Destruction (Power Management)
	FDEEEcPP20E:FCS_CKM.4(b): Cryptographic Key Destruction (TOE-Controlled Hardware)
	FDEAAcPP20E:FCS_CKM.4(d): Cryptographic Key Destruction (Software TOE, 3rd Party Storage) - per TD0766
	FDEAAcPP20E:FCS_CKM_EXT.4(a): Cryptographic Key and Key Material Destruction (Destruction Timing)
	FDEEEcPP20E:FCS_CKM_EXT.4(a): Cryptographic Key and Key Material Destruction (Destruction Timing)
	FDEAAcPP20E:FCS_CKM_EXT.4(b): Cryptographic Key and Key Material Destruction (Power Management)
	FDEEEcPP20E:FCS_CKM_EXT.4(b): Cryptographic Key and Key Material Destruction (Power Management)
	FDEEEcPP20E:FCS_CKM_EXT.6: Cryptographic Key Destruction Types
	FDEAAcPP20E:FCS_COP.1(a): Cryptographic Operation (Signature Verification)
	FDEEEcPP20E:FCS_COP.1(a): Cryptographic Operation (Signature Verification)
	FDEAAcPP20E:FCS_COP.1(b): Cryptographic operation (Hash Algorithm)
	FDEEEcPP20E:FCS_COP.1(b): Cryptographic Operation (Hash Algorithm)
	FDEAAcPP20E:FCS_COP.1(c): Cryptographic operation (Keyed Hash Algorithm)
	FDEEEcPP20E:FCS_COP.1(c): Cryptographic Operation (Message Authentication)
	FDEEEcPP20E:FCS_COP.1(d): Cryptographic Operation (Key Wrapping)

	FDEEEcPP20E:FCS_COP.1(f): Cryptographic Operation (AES Data Encryption/Decryption)
	FDEAAcPP20E:FCS_COP.1(g): Cryptographic Operation (Key Encryption)
	FDEAAcPP20E:FCS_KDF_EXT.1: Cryptographic Key Derivation
	FDEEEcPP20E:FCS_KDF_EXT.1: Cryptographic Key Derivation
	FDEAAcPP20E:FCS_KYC_EXT.1: Key Chaining (Initiator)
	FDEEEcPP20E:FCS_KYC_EXT.2: Key Chaining (Recipient)
	FDEAAcPP20E:FCS_PCC_EXT.1: Cryptographic Password Construct and Conditioning
	FDEAAcPP20E:FCS_RBG_EXT.1: Extended: Cryptographic Operation (Random Bit Generation)
	FDEEEcPP20E:FCS_RBG_EXT.1: Random Bit Generation
	FDEAAcPP20E:FCS_SMC_EXT.1: Submask Combining
	FDEAAcPP20E:FCS_SNI_EXT.1: Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) - per TD0760
	FDEEEcPP20E:FCS_SNI_EXT.1: Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation)
	FDEAAcPP20E:FCS_VAL_EXT.1: Validation
	FDEEEcPP20E:FCS_VAL_EXT.1: Validation
FDP: User data protection	FDEEEcPP20E:FDP_DSK_EXT.1: Protection of Data on Disk
FMT: Security management	FDEAAcPP20E:FMT_MOF.1: Management of Functions Behavior - per TD0765
	FDEAAcPP20E:FMT_SMF.1: Specification of Management Functions - per TD0767
	FDEEEcPP20E:FMT_SMF.1: Specification of Management Functions
	FDEAAcPP20E:FMT_SMR.1: Security Roles
FPT: Protection of the TSF	FDEEEcPP20E:FPT_FUA_EXT.1: Firmware Update Authentication
	FDEAAcPP20E:FPT_KYP_EXT.1: Protection of Key and Key Material
	FDEEEcPP20E:FPT_KYP_EXT.1: Protection of Key and Key Material
	FDEAAcPP20E:FPT_PWR_EXT.1: Power Saving States
	FDEEEcPP20E:FPT_PWR_EXT.1: Power Saving States
	FDEAAcPP20E:FPT_PWR_EXT.2: Timing of Power Saving States
	FDEEEcPP20E:FPT_PWR_EXT.2: Timing of Power Saving States
	FDEEEcPP20E:FPT_RBP_EXT.1: Rollback Protection
	FDEAAcPP20E:FPT_TST_EXT.1: TSF Testing
	FDEEEcPP20E:FPT_TST_EXT.1: TSF Testing
	FDEAAcPP20E:FPT_TUD_EXT.1: Trusted Update
	FDEEEcPP20E:FPT_TUD_EXT.1: Trusted Update

Table 2 TOE Security Functional Components

5.1.1 Cryptographic support (FCS)

5.1.1.1 Authorization Factor Acquisition - per TD0759 (FDEAAcPP20E:FCS_AFA_EXT.1)

FDEAAcPP20E:FCS_AFA_EXT.1.1

The TSF shall accept the following authorization factors: [
- a submask derived from a password authorization factor conditioned as defined in FCS_PCC_EXT.1,

- an external Smartcard factor that is protecting a submask that is [generated by the TOE (using the RBG as specified in FCS_RBG_EXT.1)] protected using [RSA with Key size [2048 bits]] with user presence proved by presentation of the smartcard and [an OE defined PIN],
- 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,].]. (TD0759 applied)

5.1.1.2 Timing of Authorization Factor Acquisition (FDEAAcPP20E:FCS_AFA_EXT.2)

FDEAAcPP20E:FCS_AFA_EXT.2.1
The TSF shall reacquire the authorization factor(s) specified in FCS_AFA_EXT.1 upon transition from any Compliant power saving state specified in FPT_PWR_EXT.1 prior to permitting access to plaintext data.

5.1.1.3 Cryptographic key generation (Symmetric Keys) (FDEAAcPP20E:FCS_CKM.1(b))

FDEAAcPP20E: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.

5.1.1.4 Cryptographic Key Generation (Data Encryption Key) (FDEEEcPP20E:FCS_CKM.1(c))

FDEEEcPP20E:FCS_CKM.1.1(c)
Refinement: The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation method [***generate a DEK using the RBG as specified in FCS_RBG_EXT.1***] and specified cryptographic key sizes [***256 bits***].

5.1.1.5 Cryptographic Key Destruction (Power Management) (FDEAAcPP20E:FCS_CKM.4(a))

FDEAAcPP20E:FCS_CKM.4.1(a)
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 that meets the following: a key destruction method specified in FCS_CKM.4(d).

5.1.1.6 Cryptographic Key Destruction (Power Management) (FDEEEcPP20E:FCS_CKM.4(a))

FDEEEcPP20E:FCS_CKM.4.1(a)
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 that meets the following: a key destruction method specified in FCS_CKM_EXT.6.

5.1.1.7 Cryptographic Key Destruction (TOE-Controlled Hardware) (FDEEEcPP20E:FCS_CKM.4(b))

FDEEEcPP20E: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,
- a new value of a key],
- removal of power to the memory],
For non-volatile memory [that employs a wear-leveling algorithm, the destruction shall be executed by a [block erase]]] that meets the following: no standard.

5.1.1.8 Cryptographic Key Destruction (Software TOE, 3rd Party Storage) - per TD0766 (FDEAAcPP20E:FCS_CKM.4(d))

FDEAAcPP20E: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,
- ones]],
- For non-volatile storage that consists of the invocation of an interface provided by the underlying platform that [instructs the underlying platform to destroy the abstraction that represents the key]]
that meets the following: no standard.

5.1.1.9 Cryptographic Key and Key Material Destruction (Destruction Timing) (FDEAAcPP20E:FCS_CKM_EXT.4(a))

FDEAAcPP20E:FCS_CKM_EXT.4.1(a)

The TSF shall destroy all keys and key material when no longer needed.

5.1.1.10 Cryptographic Key and Key Material Destruction (Destruction Timing) (FDEEEcPP20E:FCS_CKM_EXT.4(a))

FDEEEcPP20E:FCS_CKM_EXT.4.1(a)

The TSF shall destroy all keys and keying material when no longer needed.

5.1.1.11 Cryptographic Key and Key Material Destruction (Power Management) (FDEAAcPP20E:FCS_CKM_EXT.4(b))

FDEAAcPP20E:FCS_CKM_EXT.4.1(b)

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

5.1.1.12 Cryptographic Key and Key Material Destruction (Power Management) (FDEEEcPP20E:FCS_CKM_EXT.4(b))

FDEEEcPP20E:FCS_CKM_EXT.4.1(b)

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.

5.1.1.13 Cryptographic Key Destruction Types (FDEEEcPP20E:FCS_CKM_EXT.6)

FDEEEcPP20E:FCS_CKM_EXT.6.1

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

5.1.1.14 Cryptographic Operation (Signature Verification) (FDEAAcPP20E:FCS_COP.1(a))

FDEAAcPP20E:FCS_COP.1.1(a)

Refinement: The TSF shall perform cryptographic signature services (verification) in accordance with a [***RSA Digital Signature Algorithm with a key size (modulus) of [4096-bit]***]
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***].

5.1.1.15 Cryptographic Operation (Signature Verification) (FDEEEcPP20E:FCS_COP.1(a))

FDEEEcPP20E:FCS_COP.1.1(a)

The TSF shall perform cryptographic signature services (verification) in accordance with a [***RSA Digital Signature Algorithm with a key size (modulus) of [4096-bit]***] that meets 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***].

5.1.1.16 Cryptographic operation (Hash Algorithm) (FDEAAcPP20E:FCS_COP.1(b))

FDEAAcPP20E:FCS_COP.1.1(b)

Refinement: The TSF shall perform cryptographic hashing services in accordance with a specified cryptographic algorithm [***SHA-256, SHA-512***] that meet the following: ISO/IEC 10118-3:2004.

5.1.1.17 Cryptographic Operation (Hash Algorithm) (FDEEEcPP20E:FCS_COP.1(b))

FDEEEcPP20E:FCS_COP.1.1(b)

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

5.1.1.18 Cryptographic operation (Keyed Hash Algorithm) (FDEAAcPP20E:FCS_COP.1(c))

FDEAAcPP20E:FCS_COP.1.1(c)

Refinement: The TSF shall perform cryptographic [keyed-hash message authentication] in accordance with a specified cryptographic algorithm [***HMAC-SHA-512***] and cryptographic key sizes [**512 bits**] that meet the following: ISO/IEC 9797-2:2011, Section 7 'MAC Algorithm 2'.

5.1.1.19 Cryptographic Operation (Message Authentication) (FDEEEcPP20E:FCS_COP.1(c))

FDEEEcPP20E:FCS_COP.1.1(c)

Refinement: The TSF shall perform message authentication in accordance with a specified cryptographic algorithm [***HMAC-SHA-256***] and cryptographic key sizes [**256**] that meet the following: [***ISO/IEC 9797-2:2011, Section 7 'MAC Algorithm 2', NIST SP 800-16 38B***].

5.1.1.20 Cryptographic Operation (Key Wrapping) (FDEEEcPP20E:FCS_COP.1(d))

FDEEEcPP20E: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***].

5.1.1.21 Cryptographic Operation (AES Data Encryption/Decryption) (FDEEEcPP20E:FCS_COP.1(f))

FDEEEcPP20E: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***].

5.1.1.22 Cryptographic Operation (Key Encryption) (FDEAAcPP20E:FCS_COP.1(g))

FDEAAcPP20E:FCS_COP.1.1(g)

Refinement: Refinement: The TSF shall perform key encryption and decryption in accordance with a specified cryptographic algorithm AES used in [***GCM***] mode and cryptographic key sizes [***256 bits***] that meet the following: AES as specified in ISO /IEC 18033-3, [***as specified in ISO/IEC 19772***].

5.1.1.23 Cryptographic Key Derivation (FDEAAcPP20E:FCS_KDF_EXT.1)

FDEAAcPP20E:FCS_KDF_EXT.1.1

The TSF shall accept [***a RNG generated submask as specified in FCS_RBG_EXT.1, a conditioned password submask, imported submask***] to derive an intermediate key, as defined in [***NIST SP 800-108 [KDF in Counter Mode], NIST SP 800-132***], using the keyed-hash functions specified in FCS_COP.1(c), such that the output is at least of equivalent security strength (in number of bits) to the BEV.

5.1.1.24 Cryptographic Key Derivation (FDEEEcPP20E:FCS_KDF_EXT.1)

FDEEEcPP20E:FCS_KDF_EXT.1.1

The TSF shall accept [***a RNG generated submask as specified in FCS_RBG_EXT.1, a conditioned password submask, imported submask***] to derive an intermediate key, as defined in [***NIST SP 800-132***], using the keyed-hash functions specified in FCS_COP.1(c), such that the output is at least of equivalent security strength (in number of bits) to the BEV.

5.1.1.25 Key Chaining (Initiator) (FDEAAcPP20E:FCS_KYC_EXT.1)

FDEAAcPP20E:FCS_KYC_EXT.1.1

The TSF shall maintain a key chain of: [***- intermediate keys originating from one or more submask(s) to the BEV using the following method(s): [***
- key derivation as specified in FCS_KDF_EXT.1,
- key combining as specified in FCS_SMC_EXT.1,
- key encryption as specified in FCS_COP.1(g)]]
while maintaining an effective strength of [***256 bits***] for symmetric keys and an effective strength of [***not applicable***] for asymmetric keys.

FDEAAcPP20E:FCS_KYC_EXT.1.2

The TSF shall provide at least a [***256 bit***] BEV to [**SED**] [***without validation taking place***].

5.1.1.26 Key Chaining (Recipient) (FDEEEcPP20E:FCS_KYC_EXT.2)

FDEEEcPP20E:FCS_KYC_EXT.2.1

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

FDEEEcPP20E: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): [
- key 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.

5.1.1.27 Cryptographic Password Construct and Conditioning (FDEAAcPP20E:FCS_PCC_EXT.1)

FDEAAcPP20E:FCS_PCC_EXT.1.1

A password used by the TSF to generate a password authorization factor shall enable at least [**128**] characters in the set of upper case characters, lower case characters, numbers, and ['~', '!', '@', '#', '$', '^', '&', '*', '(', ')', '_', '-', '+', '=', '[', ']', ':', '<', '>', '.'] and shall perform Password-based Key Derivation Functions in accordance with a specified cryptographic algorithm HMAC-[***SHA-512***], with [***[111254] iterations***], and output cryptographic key sizes [**256 bit**] that meet the following: NIST SP 800-132. (TD0929 applied)

5.1.1.28 Extended: Cryptographic Operation (Random Bit Generation) (FDEAAcPP20E:FCS_RBG_EXT.1)

FDEAAcPP20E:FCS_RBG_EXT.1.1

The TSF shall perform all deterministic random bit generation services in accordance with [***[NIST SP 800-90A]***] using [***CTR_DRBG (AES)***]].

FDEAAcPP20E:FCS_RBG_EXT.1.2

The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from [***[1] software-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.

5.1.1.29 Random Bit Generation (FDEEEcPP20E:FCS_RBG_EXT.1)

FDEEEcPP20E:FCS_RBG_EXT.1.1

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

FDEEEcPP20E: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.

5.1.1.30 Submask Combining (FDEAAcPP20E:FCS_SMC_EXT.1)

FDEAAcPP20E:FCS_SMC_EXT.1.1

The TSF shall combine submasks using the following method [***SHA-256***] to generate an intermediary key or BEV.

5.1.1.31 Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) - per TD0760 (FDEAAcPP20E:FCS_SNI_EXT.1)

FDEAAcPP20E:FCS_SNI_EXT.1.1

The TSF shall [***use salts that are generated by a [DRBG as specified in FCS_RBG_EXT.1]***].

FDEAAcPP20E:FCS_SNI_EXT.1.2

The TSF shall use [***no nonces***].

FDEAAcPP20E:FCS_SNI_EXT.1.3

The TSF shall [***create IVs in the following matter [GCM: IV shall be non-repeating. The number of invocations of GCM shall not exceed 2^32 for a given secret key]***]. (TD0760 applied)

5.1.1.32 Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) (FDEEEcPP20E:FCS_SNI_EXT.1)

FDEEEcPP20E:FCS_SNI_EXT.1.1

The TSF shall use [***use salts that are generated by a [DRBG as specified in FCS_RBG_EXT.1]***].

FDEEEcPP20E:FCS_SNI_EXT.1.2

The TSF shall use [***unique nonces with a minimum size of 64 bits***].

FDEEEcPP20E:FCS_SNI_EXT.1.3

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***].

5.1.1.33 Validation (FDEAAcPP20E:FCS_VAL_EXT.1)

FDEAAcPP20E:FCS_VAL_EXT.1.1

The TSF shall perform validation of the [***intermediate key***] using the following methods: [***decrypt a known value using the [intermediate key] as specified in FCS_COP.1(f) and compare it against a stored known value***].

FDEAAcPP20E:FCS_VAL_EXT.1.2

The TSF shall require validation of the BEV prior to forwarding the BEV to the EE.

FDEAAcPP20E:FCS_VAL_EXT.1.3

The TSF shall [***perform a key sanitization of the DEK upon a [configurable number] of***

consecutive failed validation attempts, require power cycle/reset the TOE after [5] of consecutive failed validation attempts].

5.1.1.34 Validation (FDEEEcPP20E:FCS_VAL_EXT.1)

FDEEEcPP20E:FCS_VAL_EXT.1.1
The TSF shall perform validation of the BEV using the following method(s): [***decrypt a known value using the [BEV] as specified in FCS_COP.1(f) and compare it against a stored known value***]

FDEEEcPP20E: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.

FDEEEcPP20E:FCS_VAL_EXT.1.3
The TSF shall [***require power cycle/reset the TOE after [5] of consecutive failed validation attempts***]

5.1.2 User data protection (FDP)

5.1.2.1 Protection of Data on Disk (FDEEEcPP20E:FDP_DSK_EXT.1)

FDEEEcPP20E: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.

FDEEEcPP20E:FDP_DSK_EXT.1.2
The TSF shall encrypt all protected data without user intervention.

5.1.3 Security management (FMT)

5.1.3.1 Management of Functions Behavior - per TD0765 (FDEAAcPP20E:FMT_MOF.1)

FDEAAcPP20E:FMT_MOF.1.1
The TSF shall restrict the ability to [modify the behaviour of] the functions [use of Compliant power saving state] to [authorized users].

5.1.3.2 Specification of Management Functions - per TD0767 (FDEAAcPP20E:FMT_SMF.1)

FDEAAcPP20E:FMT_SMF.1.1
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***]
(TD0767 applied)

5.1.3.3 Specification of Management Functions (FDEEEcPP20E:FMT_SMF.1)

FDEEEcPP20E:FMT_SMF.1.1
The TSF shall be capable of performing the following management functions: a) change the DEK, as specified in FCS_CKM.1, when reprovisioning or when commanded, b) erase the DEK, as specified in FCS_CKM.4(a), c) initiate TOE firmware/software updates, d) [***no other functions***].

5.1.3.4 Security Roles (FDEAAcPP20E:FMT_SMR.1)

FDEAAcPP20E:FMT_SMR.1.1
The TSF shall maintain the roles [authorized user].

FDEAAcPP20E:FMT_SMR.1.2
The TSF shall be able to associate users with roles.

5.1.4 Protection of the TSF (FPT)

5.1.4.1 Firmware Update Authentication (FDEEEcPP20E:FPT_FUA_EXT.1)

FDEEEcPP20E: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) using the RTU that contains [***hash value of the public key as specified in FCS_COP.1(b)***].

FDEEEcPP20E: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).

FDEEEcPP20E: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 as described in FPT_TUD_EXT.1.2.

FDEEEcPP20E:FPT_FUA_EXT.1.4
The TSF shall return an error code if any part of the firmware update process fails.

5.1.4.2 Protection of Key and Key Material (FDEAAcPP20E:FPT_KYP_EXT.1)

FDEAAcPP20E:FPT_KYP_EXT.1.1
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)***].

5.1.4.3 Protection of Key and Key Material (FDEEEcPP20E:FPT_KYP_EXT.1)

FDEEEcPP20E:FPT_KYP_EXT.1.1
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)***].

5.1.4.4 Power Saving States (FDEAAcPP20E:FPT_PWR_EXT.1)

FDEAAcPP20E:FPT_PWR_EXT.1.1
The TSF shall define the following Compliant power saving states: [***G3***].

5.1.4.5 Power Saving States (FDEEEcPP20E:FPT_PWR_EXT.1)

FDEEEcPP20E:FPT_PWR_EXT.1.1
The TSF shall define the following Compliant power saving states: [***D3***].
(TD0464 applied)

5.1.4.6 Timing of Power Saving States (FDEAAcPP20E:FPT_PWR_EXT.2)

FDEAAcPP20E:FPT_PWR_EXT.2.1
For each Compliant power saving state defined in FPT_PWR_EXT.1.1, the TSF shall enter the Compliant power saving state when the following conditions occur: user-initiated request, [***system shutdown***].

5.1.4.7 Timing of Power Saving States (FDEEEcPP20E:FPT_PWR_EXT.2)

FDEEEcPP20E:FPT_PWR_EXT.2.1
For each Compliant power saving state defined in FPT_PWR_EXT.1.1, the TSF shall enter the Compliant power saving state when the following conditions occur: user-initiated request, [***system shutdown***].

5.1.4.8 Rollback Protection (FDEEEcPP20E:FPT_RBP_EXT.1)

FDEEEcPP20E:FPT_RBP_EXT.1.1
The TSF shall verify that the new firmware package is not downgrading to a lower security version number by [**inspecting the security version defined in old the Firmware and comparing that to the security version in the new**].

FDEEEcPP20E:FPT_RBP_EXT.1.2
The TSF shall generate and return an error code if the attempted firmware update package is detected to be an invalid version.

5.1.4.9 TSF Testing (FDEAAcPP20E:FPT_TST_EXT.1)

FDEAAcPP20E:FPT_TST_EXT.1.1
The TSF shall run a suite of the following self-tests [***during initial start-up (on power on), at the conditions before the function is first invoked***] to demonstrate the correct operation of the TSF: [
Power up tests
Integrity Test: Crypto library,
AES CBC 256 bit Encrypt + Decrypt KATs,
SHA-256/384/512 KAT,
RSA sign/verify KAT,
HMAC-SHA-256/384/512 KAT,
DRBG KAT,
DRBG Health Tests.
Conditional tests:
DRBG Health Tests,
Software Upgrade verification: RSA Signature Verification].

5.1.4.10 TSF Testing (FDEEEcPP20E:FPT_TST_EXT.1)

FDEEEcPP20E:FPT_TST_EXT.1.1
The TSF shall run a suite of the following self-tests [***during initial start-up (on power on), at the conditions before the function is first invoked***] to demonstrate the correct operation of the TSF: [
Power up tests
Boot code SHA KAT,
Boot code RSA KAT,
Boot code Integrity Test: Signature Verification,
Firmware Integrity Test: Signature Verification,
Firmware AES XTS 256 bit Encrypt + Decrypt KATs,
Firmware AES CBC 256 bit Encrypt + Decrypt KATs,
Firmware AES ECB 256 bit Decrypt + Decrypt KATs,
Firmware SHA-512 KAT,
Firmware SHA-256 KAT,
Firmware RSA-4096 SHA-512 TEXTBOOK KAT,
Firmware RSA-4096 SHA-512 PKCS1 V2-1 KAT,
Firmware HMAC-SHA-256 KAT,
Firmware HMAC-SHA-512 KAT,
Firmware AES Key Wrap KAT,
Firmware AES Key Unwrap KAT,
Firmware PBKDF KAT,

Firmware DRBG KAT,
Firmware DRBG Health Tests.
Conditional tests:
Firmware DRBG Health Tests,
Firmware Download Integrity Check: Signature Verification.].

5.1.4.11 Trusted Update (FDEAAcPP20E:FPT_TUD_EXT.1)

FDEAAcPP20E:FPT_TUD_EXT.1.1
Refinement: The TSF shall provide authorized users the ability to query the current version of the TOE [***software***].

FDEAAcPP20E:FPT_TUD_EXT.1.2
Refinement: The TSF shall provide authorized users the ability to initiate updates to TOE [***software***].

FDEAAcPP20E:FPT_TUD_EXT.1.3
Refinement: The TSF shall verify updates to the TOE software using a digital signature as specified in FCS_COP.1(a) by the manufacturer prior to installing those updates.

5.1.4.12 Trusted Update (FDEEEcPP20E:FPT_TUD_EXT.1)

FDEEEcPP20E:FPT_TUD_EXT.1.1
Refinement: The TSF shall provide authorized users the ability to query the current version of the TOE [***firmware***].

FDEEEcPP20E:FPT_TUD_EXT.1.2
Refinement: The TSF shall provide authorized users the ability to initiate updates to TOE [***firmware***].

FDEEEcPP20E:FPT_TUD_EXT.1.3
Refinement: The TSF shall verify updates to the TOE [***firmware***] using a [***authenticated firmware update mechanism as described in FPT_FUA_EXT.1***] by the manufacturer prior to installing those updates.

5.2 TOE Security Assurance Requirements

The SARs for the TOE are the components as specified in Part 3 of the Common Criteria. Note that the SARs have effectively been refined with the assurance activities explicitly defined in association with both the SFRs and SARs.

Requirement Class	Requirement Component
ADV: Development	ADV_FSP.1: Basic Functional Specification
AGD: Guidance documents	AGD_OPE.1: Operational User Guidance
	AGD_PRE.1: Preparative Procedures
ALC: Life-cycle support	ALC_CMC.1: Labelling of the TOE
	ALC_CMS.1: TOE CM Coverage
ATE: Tests	ATE_IND.1: Independent Testing - Conformance
AVA: Vulnerability assessment	AVA_VAN.1: Vulnerability Survey

Table 3 Assurance Components

5.2.1 Development (ADV)

5.2.1.1 Basic Functional Specification (ADV_FSP.1)

ADV_FSP.1.1d
The developer shall provide a functional specification.

ADV_FSP.1.2d
The developer shall provide a tracing from the functional specification to the SFRs.

ADV_FSP.1.1c
The functional specification shall describe the purpose and method of use for each SFR-enforcing and SFR-supporting TSFI.

ADV_FSP.1.2c
The functional specification shall identify all parameters associated with each SFR-enforcing and SFR-supporting TSFI.

ADV_FSP.1.3c
The functional specification shall provide rationale for the implicit categorization of interfaces as SFR-non-interfering.

ADV_FSP.1.4c
The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.

ADV_FSP.1.1e
The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.1.2e
The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

5.2.2 Guidance documents (AGD)

5.2.2.1 Operational User Guidance (AGD_OPE.1)

AGD_OPE.1.1d
The developer shall provide operational user guidance.

AGD_OPE.1.1c
The operational user guidance shall describe, for each user role, the user-accessible functions and privileges that should be controlled in a secure processing environment, including appropriate warnings.

AGD_OPE.1.2c
The operational user guidance shall describe, for each user role, how to use the available interfaces provided by the TOE in a secure manner.

AGD_OPE.1.3c
The operational user guidance shall describe, for each user role, the available functions and interfaces, in particular all security parameters under the control of the user, indicating secure values as appropriate.

AGD_OPE.1.4c
The operational user guidance shall, for each user role, clearly present each type of security-relevant event relative to the user-accessible functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

AGD_OPE.1.5c
The operational user guidance shall identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences, and implications for maintaining secure operation.

AGD_OPE.1.6c
The operational user guidance shall, for each user role, describe the security measures to be followed in order to fulfill the security objectives for the operational environment as described in the ST.

AGD_OPE.1.7c
The operational user guidance shall be clear and reasonable.

AGD_OPE.1.1e
The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.2.2 Preparative Procedures (AGD_PRE.1)

AGD_PRE.1.1d
The developer shall provide the TOE, including its preparative procedures.

AGD_PRE.1.1c
The preparative procedures shall describe all the steps necessary for secure acceptance of the delivered TOE in accordance with the developer's delivery procedures.

AGD_PRE.1.2c
The preparative procedures shall describe all the steps necessary for secure installation of the TOE and for the secure preparation of the operational environment in accordance with the security objectives for the operational environment as described in the ST.

AGD_PRE.1.1e
The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AGD_PRE.1.2e
The evaluator shall apply the preparative procedures to confirm that the TOE can be prepared securely for operation.

5.2.3 Life-cycle support (ALC)

5.2.3.1 Labelling of the TOE (ALC_CMC.1)

ALC_CMC.1.1d
The developer shall provide the TOE and a reference for the TOE.

ALC_CMC.1.1c
The TOE shall be labelled with its unique reference.

ALC_CMC.1.1e
The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.3.2 TOE CM Coverage (ALC_CMS.1)

ALC_CMS.1.1d
The developer shall provide a configuration list for the TOE.

ALC_CMS.1.1c
The configuration list shall include the following: the TOE itself; and the evaluation evidence required by the SARs.

ALC_CMS.1.2c
The configuration list shall uniquely identify the configuration items.

ALC_CMS.1.1e
The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.4 Security Target (ASE)

5.2.4.1 Cryptographic operation (Hash Algorithm) (ASE_TSS.1(c))

ASE_TSS.1.1(c)
Refinement: The TOE summary specification shall describe how the TOE meets each SFR, including a proprietary Key Management Description (Appendix E), and [**Entropy Essay**].

5.2.5 Tests (ATE)

5.2.5.1 Independent Testing - Conformance (ATE_IND.1)

ATE_IND.1.1d
The developer shall provide the TOE for testing.

ATE_IND.1.1c
The TOE shall be suitable for testing.

ATE_IND.1.1e
The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_IND.1.2e
The evaluator shall test a subset of the TSF to confirm that the TSF operates as specified.

5.2.6 Vulnerability assessment (AVA)

5.2.6.1 Vulnerability Survey (AVA_VAN.1)

AVA_VAN.1.1d
The developer shall provide the TOE for testing.

AVA_VAN.1.1c
The TOE shall be suitable for testing.

AVA_VAN.1.1e
The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VAN.1.2e
The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the TOE.

AVA_VAN.1.3e
The evaluator shall conduct penetration testing, based on the identified potential vulnerabilities, to determine that the TOE is resistant to attacks performed by an attacker possessing Basic attack potential.

6. TOE Summary Specification

This chapter describes the security functions:

- Cryptographic support
- User data protection
- Security management
- Protection of the TSF

6.1 Context

6.1.1 Supported Authorization Factors

The TOE supports the following different authorization factors (as well as a dual-factor authentication, which combines a password with one of the other methods)

1. Password – the user supplies a secret password
2. Smartcard – the user inserts their smartcard as well as the PIN needed to authenticate *to the smartcard*.
3. USB devices
    a. USB FIDO2 compliant Security Key ("Security Key" for short) – the user inserts their Security Key (and depending upon the Security Key's configuration, may need to enables it by either supplying a PIN or proving physical presence by pressing a button on the Security Key itself).
    b. USB Drive – the user inserts their USB drive and the TOE reads a secret value stored in file residing on the USB drive.

Again, for dual-factor, the TOE combines the Password method with one of the other supported "token" methods (and combines those authorization factors as described later).

In all cases, the TOE, after receiving the authorization factor(s), transforms them using PBKDFv2, validates whether the authorization factor(s) is/are correct, and if correct, attempts to unlock the SED by passing the BEV to the SED.

6.1.2 Core TOE Concepts

The following are core concepts and TOE components relevant to understanding the TSS:

a) Installer. The TOE installer runs from a universal serial bus (USB) drive used to boot the TOE and perform the drive configuration. It will accept the SED administrator password and new TOE administrator password as input, bring the SED device from factory state to functional Opal state, take ownership of the SED, enable the Shadow MBR, create the EFI system partition (ESP) and install all the TOE components. At completion of the install, the hardware platform administrator sets the new TOE partition as the first boot option in the UEFI boot option list.

b) Shadow MBR. A 128-MB read-only partition of the SED that is the only partition visible until the SED is unlocked by the TOE. Once the SED is unlocked the Shadow MBR is mapped out and the protected partitions mapped in.

c) ESP. EFI System Partition (ESP) is a GUID partition table (GPT) partition with file allocation table (FAT) FAT32 file system located in the Shadow MBR. The system firmware loads files from this partition to boot and load the TOE.

d) Database. The TOE stores an encrypted database in the Opal provided additional 'DataStore' section. This is an up to 64MB storage area that can be read/written using specific TCG Opal commands. The database will be readable by a 'datastore' user (that only has rights to read from the DataStore area). The DB will be read out of the DataStore area and stored in a temporary in-memory file. Only an authenticated user will be able to make changes (as the changes will have to be written back to the DataStore section).

e) GUI. The TOE provides a local graphical user interface (GUI) for PBA (SED unlock via authentication factor(s)) and TOE / user management.

f) User Management. The TOE enforces role-based access control with the following roles defined:

   i. Admin. Can unlock the SED, add other users and update TOE firmware.

   ii. Login User. Can unlock the SED.

g) Protected OS. The protected OS environment on the SED that is booted after successful TOE authentication.

6.1.3 Key Management

The following sections describe the fundamental key management aspects of the TOE. The Figures below depict the resulting AA and EE keychains designed with sufficient strength to protect the end EE's 256-bit data encryption key (DEK).

Key Table AA/PBA

Name	Use	Type	Source	Storage
Password	Authentication factor	Password	Input by user	Volatile memory
KEK2	Authentication factor or pre-cursor	256-bit secret	Generated by DRBG,	Non-volatile drive or external USB Drive (plaintext)
int-KEK2	Authentication factor	2048-bit or 256-bit (smartcard or USB)	Output from smartcard or USB token	Volatile memory
SUB1/2/3	Used to encrypt KEK1	256-bit AES GCM	Derived from Authentication factors	Volatile memory
KEK1	Used to encrypt AK	256-bit AES GCM	Generated by DRBG, Decrypted w/ SUBx	Volatile memory & Non-volatile (encrypted)
AK (BEV)	Authenticator factor	256-bit conditioned authentication factor	Generated by DRBG, Decrypted w/ KEK1	Volatile memory & Non-volatile (encrypted)

Key Table EE/SED

Name	Use	Type	Source	Storage
BEV	Authentication factor	256-bit conditioned authentication factor	Input from AA	Volatile memory
PB-KEK	Used to encrypt KEK	256-bit AES KW	Derived from BEV	Volatile memory
KEK	Used to encrypt DEK	256-bit AES KW	Generated by DRBG, Decrypted w/ PB-KEK	Volatile memory & Non-volatile (encrypted)
DEK	Data encryption key	256-bit AES XTS	Generated by DRBG, Decrypted w/ KEK	Volatile memory & Non-volatile (encrypted)

6.1.4 Authentication Keys

The TOE' generates and manages the Authentication Keys (AKs) used to unlock the SED (AKs are the border encryption value (BEV) referred to by the CPP_FDE_AA). The OPAL 2.0 standard specifies the following standard SED ‘user accounts’:

a) SID. Security ID – the owner of the SED (e.g. root).

b) ADMIN SP. This is the Administrative Security Provider. It is the OPAL construct that administers the security on the SED.

c) LOCKING SP. This is the Locking Security Provider. It is the OPAL construct that manages the locking and unlocking of the locking ranges on the SED.

During installation, the TOE generates a 256-bit BEV (AK). AKs may be generated for the SID, ADMIN SP and LOCKING SP SED user accounts. The AKs are encrypted using AES-GCM-256 and stored in the TOE’s database.

6.1.5 Key Chain

As show in Figures 1, 2, and 3 the TOE uses a chain of up to two key encryption keys (KEKs) to the BEV (AK), depending on the authentication mechanism:

a) SUB1. 256-bit submask derived from the user’s password via password-based key derivation function (PBKDF2).

b) SUB2. 256-bit submask derived (using PBKDF2) from the user’s smartcard’s protected credential (KEK2).

c) SUB2. SUB3. 256-bit submask obtained by combining (using SHA2-256) the two submasks created through derivation of the user’s password and smartcard credentials.

d) KEK1. The PBA randomly generates this key and encrypts a copy using the SUB_x resulting from the user authorization factors in use:

e) KEK2. The TOE generates KEK2, a 256-byte random string, using its DRBG, asks the user’s smartcard to encrypt the string, and saves the resulting encrypted blob (which the PBA will later present back to the smartcard during boot/unlock) in its database. The PBA does this for each user employing a smartcard. The user’s smartcard possesses an RSA private key that the smartcard uses to encrypt the PBA generated KEK2 value. Upon boot, the PBA provides the smartcard both the user’s PIN and the RSA encrypted KEK2 value and requests the smartcard decrypt and return the plaintext KEK2. Upon receiving back the 256-byte KEK2 string, the PBA uses PBKDF2 to derive SUB2.

f) AK (BEV): The PBA generates (using its DRBG) a random 256-bit Authentication Key for each user. The AK becomes the BEV (in the protection profiles’ parlance) which the PBA passes to the SED (or EE portion) in order to unlock the drive.

6.1.6 Authentication / Drive Unlock Flow

At a high-level, the basic start-up and authentication flow is as follows:

a. When the TOE starts up, the SED’s Shadown MBR is active, and the TOE boots from the PBA code partition, launches the PBA GUI, copies and de-obfuscates the database from the PBA data partition and mounts it in random access memory (RAM). The user then enters their username and password, inserts a physical token (a token can be a smartcard, a USB FIDO2 compliant Security Key, or a USB drive), or a combination of password and token.

b. Depending on the authentication method:

    4. For password-only, the TOE validates the username against the database.

    5. For smartcard-only, the smartcard PIN is authenticated.

6. For a USB FIDO2 compliant Security Key only (AKA "Security Key"), the security key PIN is authenticated (if enabled) or presence confirmed (through a button push).
7. For a USB Drive, the TOE validates the authentication factor/key
8. For dual-factor, the TOE validates the username against the database, and validates the token (which acts as secondary authentication factor, i.e., a smartcard or security key).

c. The TOE performs PBKDF2 on the password to generate SUB1.

d. The TOE performs PBKDF2 on the physical token output to generate SUB2.

e. The TOE combines SUB1 and SUB2 (SHA-256) to form SUB3 if employing dual-factor authentication. SUB3 is then used to encrypt/decrypt the KEK using AES-256-GCM.

f. The TOE uses the plaintext KEK value to decrypt the AK.

g. The TOE establishes an authenticated session with the opal subsystem.

h. The TOE transitions the Shadow MBR off.

i. The TOE unlocks the global range.

j. The TOE initiates a soft reboot, allowing the TOE OS (that is now accessible as the Shadow MBR, transitioned off, and the global range has been unlocked) to boot.

k. If the unlock succeeds, the user is authenticated / authorized and the TOE sets mbrdone to true and unlocks the global range. If the drive specific retry attempts are exhausted, the system will be shut down.

6.2 Cryptographic support

FDEAAcPP20E:FCS_AFA_EXT.1:

The TOE supports the use of password-only, smartcard-only, USB FIDO2 compliant Security Key only, USB drive only, and dual factor authentication (which combines a username/password with a physical token: either a smartcard or a USB Security Key).

The TOE supports an external smartcard factor that is at least the same bit-length as the DEK (256-bit) – the KEK2 value released by the smartcard is 2048-bits in length. The TOE uses a standard PC/SC Interface to communicate with a Smartcard and obtain the smartcard authentication factor.

The TOE also supports a FIDO2 security key that can function in one of three different access methods (a PIN [similar to smartcard operation], a physical button to confirm presence, or none).

Finally, the TOE supports an external USB drive that houses a plaintext copy of authentication factor KEK2.

FDEAAcPP20E:FCS_AFA_EXT.2:

The TOE supports both passwords, a credential for the smartcard or the USB token, and the presence of a USB drive (or a combination of password and smartcard or security key, referred to as "dual factor" authentication). The TOE requires the user (re)present his or her authentication factor(s) after a power cycle (power off followed by power on).

FDEAAcPP20E:FCS_CKM.1(b):

The TOE's PBA uses a 256-bit Authentication Key (AK) or BEV value. The PBA also stores the AK encrypted with KEK1 (which is also a 256-bit AES GCM key), and in turn, stores a copy of the KEK1 GCM encrypted with a key derived from each user's authentication factor (SUB1, SUB2, SUB3, all 256-bit AES GCM keys).

FDEEEcPP20E:FCS_CKM.1(c):

The TOE's SED generates 256-bit AES-XTS key Data Encryption Key (DEK) and 256-bit AES-GCM KEKs (per user) using its SHA2-256 HMAC_DRBG within the SED. The TOE accesses its internal HMAC_DRBG whenever keys or IVs are needed. The TOE does not generate DEKs outside of it TOE boundary nor does it receive a DEK from outside.

FDEAAcPP20E:FCS_CKM.4(a):

The TOE's PBA erases cryptographic keys and key material from volatile memory when transitioning to a Compliant power saving state with a single overwrite consisting of zeroes and a second, single, overwrite consisting of ones as specified in FCS_CKM.4(d).

Note: The TSF (not the Operational Environment) is used to destroy keys from volatile memory.

FDEEEcPP20E:FCS_CKM.4(a):

The TOE's SED erases cryptographic keys and key material from volatile memory when the transitioning to the power-off states with a single overwrite consisting of zeros, or alternatively overwrite with a new key value, or finally, with a removal of power to the memory as specified in FCS_CKM_EXT.6 and FCS_CKM_EXT.4(b).

FDEEEcPP20E:FCS_CKM.4(b):

The TOE's SED introduces cryptographic keys into RAM (volatile memory) both when deriving a user's PB-KEK and when decrypting the user's KEK or the DEK. When the SED no longer needs these values (after use), the SED overwrites the key memory with zeros.

Additionally, the TOE's SED allows an administrator to update a key value (which destructively resets the data encrypted by that key). In this manner, the TOE allows clearing of a key value via an overwrite of a new key value.

Finally, the TOE's SED will experience destruction of all keys in RAM upon removal of power to the drive/SED.

The TOE's SED only stored encrypted keys persistently in the memory controller's NOR/NAND flash memory. The SED uses a block erase to destroy all keys stored in non-volatile memory.

FDEAAcPP20E:FCS_CKM.4(d):

For the TOE's PBA volatile memory, key destruction is executed by a single overwrite consisting of zeroes followed by a single overwrite of ones, immediately following the operation requiring the key is completed.

For the TOE's PBA non-volatile memory, the TOE GUI may be used to forward requests to cryptographically erase the DEK to the encryption engine (EE) by uninstalling the TOE or erasing the entire disk. On the admin's request, the SED will crypto erase itself (internally, the PBA sends a Opal Revert Tper command to the drive followed immediately by a crypto erase [format nvm]).

Additional details regarding how keys are managed in volatile and non-volatile memory are provided in the Key Management Description (KMD).

FDEAAcPP20E:FCS_CKM_EXT.4(a):

At a high level, the TOE's PBA keys are no longer needed when power is removed from memory, when the TOE erases the disk, or when the TOE is uninstalled. All intermediate keys are destroyed after their use in the chain. For example, for password-only authentication, the user's SUB1 is destroyed subsequent to the decryption of the KEK1 and the AK

FDEEEcPP20E:FCS_CKM_EXT.4(a):

At a high level, the TOE's SED keys are no longer needed when power is removed or when the SED receives instruction to erase itself. The TOE destroys all intermediate keys after user. Additional details regarding timing of key destruction are provided in the KMD.

FDEAAcPP20E:FCS_CKM_EXT.4(b):

Transitioning into the compliant power saving state automatically triggers the destruction of all keys and keying material from volatile memory.

Additional details regarding key destruction when entering a Compliant power saving state are provided in the KMD.

FDEEEcPP20E:FCS_CKM_EXT.4(b):

Transitioning into the compliant power saving state automatically triggers the destruction of all keys and keying material from volatile memory.

Additional details regarding key destruction when entering a Compliant power saving state are provided in the KMD.

FDEEEcPP20E:FCS_CKM_EXT.6:

The TOE's SED subjects plaintext keys in RAM (volatile memory) to clearing through overwriting the memory location with zeros (when no longer needed), through overwrite with a new key value, or through removal of power to the memory. Similarly, the TOE's SED provides destruction of persistently stored (non-volatile memory) encrypted keys through a block erase.

FDEAAcPP20E:FCS_COP.1:

The TOE PBA's incorporates (and executes within) a hardened Ubuntu 24.04 operating system (tested on a Dell Precision 3591 laptop with an Intel Core Ultra 7 155H (Series 1/Meteor Lake) CPU) uses the following cryptographic algorithms in support of its SW/PBA (AA) functionality.

SFR	Algorithm	NIST Standard	Cert#
FCS_COP.1(a) (Verify)	RSA 4096 Signature Verification	FIPS 186-4, RSA	A7050
FCS_COP.1(b) (Hash)	SHA2-256/512 Hashing	FIPS 180-4	A7050
FCS_COP.1(c) (Keyed Hash)	HMAC-SHA2-512	FIPS 198-1 & 180-4	A7050
FCS_COP.1(g) (AES) / FCS_CKM.1(b)	AES-256 GCM Encrypt/Decrypt	FIPS 197	A7050
FCS_RBG_EXT.1 (Random)	AES-256 CTR_DRBG	SP 800-90A	A7050

Table 4 SW-FDE (AA) PBA Cryptographic Algorithms

The TOE also uses its Cigent PS5021-E21T Module V1.00 executing within the PS5021-E21T's ARM Cortex-R5 controller in the Self-Encrypting Drive for the Encryption Engine (EE) functionality.

SFR	Algorithm	NIST Standard	Cert#
FCS_COP.1(a) (Verify)	RSA 4096 Signature Verification	FIPS 186-4, RSA	A7299
FCS_COP.1(b) (Hash)	SHA2-256/512 Hashing	FIPS 180-4	A7299
FCS_COP.1(c) (Keyed Hash)	HMAC-SHA2-256	FIPS 198-1 & 180-4	A7299
FCS_COP.1(d) (AES)	AES-256 KW Encrypt/Decrypt	FIPS SP 800-38F	A7299
FCS_COP.1(f)/ FCS_CKM.1(c) (AES)	AES-256 XTS Encrypt/Decrypt	FIPS 197	A7299
FCS_RBG_EXT.1 (Random)	SHA2-256 HMAC_DRBG	SP 800-90A	A7299

Table 5 HW-SED (EE) Cryptographic Algorithms

FDEAAcPP20E:FCS_COP.1(a):

The TOE's PBA performs signature verification using RSA 4096 with SHA-512 for trusted updates as follows:

a) TOE updates are signed with the Cigent code signing private key
b) The obfuscated public key is embedded in the TOE binary
c) When the user triggers the TOE update, the TOE verifies the digital signature using the embedded public key
d) If the digital signature verification succeeds, the upgrade process is carried out
e) If the digital signature verification fails, the upgrade process is aborted, and an error is displayed to the user.

FDEEEcPP20E:FCS_COP.1(a):

The TOE's SED performs RSA Digital Signature Algorithm verification with a key size (modulus) of 4096 bits with SHA-512 for hashing. The function complies with FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS. The SED does not generate RSA keys. RSA Digital Signature Algorithm verification is used to verify updates to the TOE's SED firmware.

FDEAAcPP20E:FCS_COP.1(b):

The TOE's PBA makes use of SHA-512 for the following:

a) Digital signature verification

b) PBKDF

The TOE's PBA makes use of SHA-256 for Submask Combining as defined in FCS_SMC_EXT.1.

FDEEEcPP20E:FCS_COP.1(b):

The TOE's SED performs SHA-256 cryptographic hashing services that meet the following: ISO/IEC 10118-3:2004. The TOE uses SHA-256 with HMAC-SHA-256 as part of the HMAC_DRBG function. The TOE also uses the SHA-512 hash functions as part of the RSA signature verification function

FDEAAcPP20E:FCS_COP.1(c):

The TOE's PBA implements HMAC-SHA-512 with the following characteristics:

a) Key length. 512 bits.

b) Block size. 1024 bits.

c) MAC length. 512 bits.

FDEEEcPP20E:FCS_COP.1(c):

The TOE's SED performs HMAC-SHA-256 message authentication using cryptographic key sizes 256 bit that meet the following: ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2". The block size is 64 bytes and the output MAC length size is 32 bytes.

FDEEEcPP20E:FCS_COP.1(d):

The TOE's SED performs AES Key Wrap per SP 800-38F. The inputs to the AES-256 Key Wrap function are shown in the table below. The output of the AES-256 Key Wrap function is a wrapped key or intermediate key.

The TOE wraps (AES KW encrypts) the plaintext DEK using the KEK. Next the TOE wraps (again, AES KW encrypts) the plaintext KEK using the PBKDF-KEK derived from the authentication factor. The TOE performs AES KW mode encryption using a key size of 256-bits that meet the following: AES as specified in ISO /IEC 18033-3.

AES Wrap Function	Key	Input (Data)	Output (Data)
1st AES Wrap	KEK	Plaintext DEK	Encrypted DEK
2nd AES Wrap	PBKDF-KEK	Plaintext KEK	Encrypted KEK

The TOE's SED performs AES Key Unwrap per SP 800-38F. The inputs to the AES-256 Key Unwrap are shown in the table below. When an administrator or user enters a PIN (as part of setting up the SED, the PBA software replaces the SED's default PIN with an administrator specified PIN), the TOE validates the PIN by first calling the PBKDF function with the PIN and the associated plaintext salt value as inputs. The output of the PBKDF function is the ephemeral plaintext PBKDF-KEK associated with that PIN. After validation, the TOE's SED using the PBKDF-KEK to AES-KW unwrap an integrity check value (termed the encrypted password or ePassword). A successful unwrap function will result in the correct integrity check value for the KW function (ICV), indicating that the PIN is valid and authentication is successful. Otherwise, the PIN is invalid and authentication is unsuccessful. The complete list of PINs (authorization factors), otherwise known as credentials is in the KMD.

If authentication succeeds, the TOE's SED uses the PBKDF-KEK to unwrap (AES-KW decrypt) the KEK, and subsequently uses the KEK to unwrap (AES-KW decrypt) the DEK.

The TOE performs AES KW mode encryption using a key size of 256-bits that meet the following: AES as specified in ISO /IEC 18033-3.

AES Unwrap Function	Key	Input (Data)	Output (Data)
1st AES Unwrap	PBKDF-KEK	Encrypted KEK	Plaintext KEK
2nd AES Unwrap	KEK	Encrypted DEK	Plaintext DEK

FDEEEcPP20E:FCS_COP.1(f):

The TOE's SED uses AES-256 XTS (as specified in IEEE 1619) to encrypt drive data.

FDEAAcPP20E:FCS_COP.1(g):

The TOE's PBA uses AES-256 bit GCM keys to protect persistently stored keys in the keychain.

FDEAAcPP20E:FCS_KDF_EXT.1:

The TOE's PBA conditions passwords via PBKDF2 using HMAC-SHA-512 with 111,254 iterations, resulting in a 256-bit key in accordance with National Institute of Standards and Technology (NIST) special publication (SP) 800-132. For smartcard authentication, the TOE accepts an RNG generated submask in accordance with NIST SP 800-108.

FDEEEcPP20E:FCS_KDF_EXT.1:

The TOE's SED conditions the BEV via PBKDF2 using HMAC-SHA-256 with 1,000 iterations, resulting in a 256-bit key in accordance with National Institute of Standards and Technology (NIST) special publication (SP) 800-132.

FDEAAcPP20E:FCS_KYC_EXT.1:

The TOE supports a BEV (AK) size of 256 bits. Additional details on the TOE key chain are provided in the KMD.

FDEEEcPP20E:FCS_KYC_EXT.2:

The TOE's SED accepts BEVs of 256 bits from the AA, maintaining a chain of intermediate keys originating from the BEV to the DEK and using the following methods:

a) key derivation as specified in FCS_KDF_EXT.1: The TOE's SED derives a PBKDF-KEK with PBKDF and the Border Encryption Value (BEV) passed in by AA/PBA,

b) key wrapping as specified in FCS_COP.1(d): The TOE unwraps (AES-KW decryption) the KEK with PBKDF-KEK and then uses the KEK to unwraps (again, AES-KW decryption) the DEK.

c) the TOE uses the XTS-AES-256 DEK to decrypts reads of disk data and encrypt writes to disk.

The TOE maintains an effective strength of 256 bits for symmetric keys

FDEAAcPP20E:FCS_PCC_EXT.1:

The TOE implements a configurable password policy with the following options:

a) Minimum Length (8 – 128)

b) Require at least one uppercase

c) Require at least one lowercase

d) Require at least one numeric

e) Require at least one of the following special characters: ("~", "!", "@", "#", "$", "^", "&", "*", "(", ")", "_", "-", "+", "=", "[", "]", ":", "<", ">", ".")

Passwords are conditioned via PBKDF2 using HMAC-SHA-512 with 111,254 iterations, resulting in a 256-bit key in accordance with NIST SP 800-132.

FDEAAcPP20E:FCS_RBG_EXT.1:

The TOE uses a software-based random bit generator (AES-256 CTR_DRBG) that complies with NIST SP 800-90A for all cryptographic operations. The DRBG is seeded by a single software-based entropy/noise source from a Jitterentropy entropy source. All entropy is extracted, processed, and accumulated by OpenSSL.

The expected amount of entropy received from the Jitter software noise source provides a 256-bit seed with a min-entropy of 1 bit per bit (or 8 bits per byte).

FDEEEcPP20E:FCS_RBG_EXT.1:

The TOE performs all deterministic random bit generation services in accordance with NIST SP 800-90A using HMAC_DRBG (any) and SHA2-256 cryptographic hashing services.

The SSD SEDs use one hardware entropy source to seed the RBG 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.

FDEAAcPP20E:FCS_SMC_EXT.1:

As described before, the TOE’s PBA combines the SUB1 and SUB2 using SHA-256 which produces SUB3, which the PBA uses to encrypt/decrypt the 256-bit KEK1.

FDEAAcPP20E:FCS_SNI_EXT.1:

PBA Salts and IVs are generated using the RBG as described in FCS_RBG_EXT.1.

The TOE’s PBA does not make use of nonces.

FDEEEcPP20E:FCS_SNI_EXT.1:

The TOE’s SED uses randomly generated 256 bit salt values using an RBG described in FCS_RBG_EXT.1 as inputs to the Password Based Key Derivation (PBKDF) function. There is a 256 bit salt value associated with each PIN value in the drive.

The tweak values used for XTS are non-negative integers, assigned consecutively, and starting at an arbitrary non-negative integer. The TOE’s SED does not use nonces or IV values.

FDEAAcPP20E:FCS_VAL_EXT.1:

The TOE’s PBA accepts authorization factors from users and validates them by PBKDFv2 deriving an AES-GCM decryption key and then attempting to decrypt the stored KEK1 key. If the AES-GCM decryption fails, the TOE rejects the user’s authorization attempt and increments the number of consecutive failed authentication attempts. When the number of failed attempts exceeds a configurable value (default of 5), the TOE forces a system restart. Additionally, one can optionally configure the TOE to erase the drive (using both a TCG Opal cryptographic erase followed by a block level erasure using formatnvm) after exceeding the failed attempt limit.

FDEEEcPP20E:FCS_VAL_EXT.1:

The TOE’s SED uses PINs (BEVs) as authentication/authorization factors and performs validation of those BEV. That said, because the PBA software lies between the human operator and the SED and because the PBA performs its own validation of authorization factors, the PBA obviates the need for any SED validation. Put another way, the PBA’s validation of authorization factors ensures that SED will never receive an incorrect BEV value. The PBA shields the SED from encountering an incorrect BEV. The remainder of this section describes the how the SED validates BEV only for the sake of completeness.

The PINs (BEVS) that the TOE’s SED accepts are not stored by the SED. Instead for each PIN (BEV), the PIN is validated by first calling the PBKDF function with the PIN and the associated plaintext salt value as inputs. The output of the PBKDF function is the ephemeral plaintext authentication key associated with that PIN. The second call is the AES-KW unwrap function with the Authentication Key. If unwrap function check results in the correct integrity check value for the KW function (ICV), the PIN is valid and authentication is successful, else the PIN is invalid and authentication is unsuccessful. The complete list of PINs (authorization factors), otherwise known as credentials is in Table 10 below. The TOE requires the validation of the BEV prior to allowing access to TSF data after exiting a compliant power saving state.

The TOE’s SED maintains a separate failure count for each PIN that keeps track of the number of failed authentication attempts. The counter is reset to zero after a successful authentication. The persistence settings are set in the factory and are not configurable.

The following table identifies the failure count maximum values, persistence and configuration options for each PIN type.

All versions of the TOE implement the same cryptographic module within the PS5021-E21T hardware controller.

Credential Name	Type	Try Limit	Try Limit Settable	Persistent
SID	TCG Opal	5 retries	NO	NO
PSID	TCG Opal	5 retries	NO	NO
Locking SP Admin 1-4 Passwords (BEV)	TCG Opal	5 retries	NO	NO
Admin SP Admin 1-4 Passwords (BEV)	TCG Opal	5 retries	NO	NO
User 1-15 Passwords (BEV)	TCG Opal	5 retries	NO	NO

6.3 User data protection

FDEEEcPP20E:FDP_DSK_EXT.1:

The TOE’s SED is encrypted by default without user intervention using AES:XTS (as described in Section 6.2). There is no restriction on reading or writing data to the SED until a user takes ownership using a TCG controller. Taking ownership locks a drive and constitutes the initialization process providing data-at-rest protection. A locked drive restricts data reads and writes based on the settings of Locking SP Users (TCG Opal).

There are three categories of storage: unencrypted for OS use, unencrypted for drive use, and encrypted. On Opal SEDs, unencrypted for OS use includes shadow MBR, which is used for boot. On an Opal SED, the system area of disk is not encrypted.

There is no host access to the system area. TCG Data Store tables are available unencrypted in the system area.

Administrators can store data in these tables through access-controlled TCG commands. The SED places no restriction on what data is stored; however, TOE’s PBA does not store any protected data in the tables.

6.4 Security management

FDEAAcPP20E:FMT_MOF.1:

The TOE’s PBA does not allow any modification related to power saving states.

FDEAAcPP20E:FMT_SMF.1:

The TOE’s PBA GUI may be used to forward requests to cryptographically erase the DEK to the encryption engine (EE or SED) via the GUI by uninstalling the TOE or erasing the entire disk. On the admin’s request, the Opal Revert Tper command is sent to the drive followed immediately by a crypto erase (format nvm).

Note: Changing the DEK is the same functionality as cryptographically erasing the DEK.

The TOE’s PBA GUI may be used to configure the authorization factors (password-only, token-only, or a combination of password + smartcard/security key).

TOE updates may be performed by booting the host system with a USB drive that contains the PBA OS, utility, and the updated PBA content. An admin user must authenticate and choose to install the update.

FDEEEcPP20E:FMT_SMF.1:

The TOE is 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).

c) Initiate TOE firmware/software update.

The TOE changes a DEK when re-provisioning or when commanded. The SED generates each DEK on the drive by using the drive's SP 800-90A HMAC Based DRBG (256 bits).

DEK destruction is described in Section 6.2.2: Cryptographic Key Destruction.

Firmware updates are initiated through the PBA (which internally uses the SED's NVMe Firmware Image Download/Firmware Image Commit Command).

To perform a firmware download, an operator performs the following steps:

1) The signed firmware package is downloaded to the drive. It is received by the drive firmware and placed into volatile memory.
2) The TOE compares a hash of the public key with the stored hash of the public key, and then verifies the digital signature.
3) The signature is verified using PKCS #1, v2.1 RSA signature algorithm and public key in ROM. If the verification fails an error is returned and the update is not performed. The RSA key/modulus size for all current generation Cigent SED products is 4096 bits.
4) The firmware update package is written to non-volatile memory. This overwrites the original firmware.
5) The FW performs a soft reset which loads and runs the new firmware.

FDEAAcPP20E:FMT_SMR.1:

The TOE restricts access to authorized users.

6.5 Protection of the TSF

FDEEEcPP20E:FPT_FUA_EXT.1:

To perform a firmware download, an administrator performs the following steps:

1) Obtain a genuine Cigent Secure firmware update package from Cigent's support web portal
2) The signed firmware package is downloaded to the drive. It is received by the drive firmware and placed into volatile memory.
3) The signature is verified using PKCS #1, v2.1 RSA signature algorithm and public key in ROM. If the verification fails an error is returned and the update is not performed. The RSA key/modulus size for all current generation Cigent products is 4096 bits.
4) The firmware update package is written to non-volatile memory. This overwrites the original firmware.
5) The FW performs a soft reset which loads and runs the new firmware.

An error code is returned if any part of the firmware update process fails. The TOE only allows installation of an update if the digital signature has been successfully verified.

The firmware key store and the signature verification algorithm are stored in a write protected area on the TOE. The firmware can only be updated using the authenticated update mechanism by an authorized user where the authorized source that signs TOE updates is Cigent. The TOE authenticates the source of the firmware update using the RSA digital signature algorithm, with a key size (modulus) of 4096 bits. The mechanism uses the Root of Trust for Update RTU key stored in ROM that contains the hashed public key. This key will be used for verifying with the public key that comes along with the signed firmware (FW) binary file. After successful verification, the public key in the binary will then be used to verify the signature on an update image. An error code is returned if any part of the firmware update process fails. The TOE only allows installation of an update if the digital signature has been successfully verified.

FDEAAcPP20E:FPT_KYP_EXT.1:

Keys are protected as described in the KMD. The AK is encrypted as per FCS_COP.1(g) and stored in the TOE's database.

FDEEEcPP20E:FPT_KYP_EXT.1:

The TOE stores all keys in non-volatile memory only when the keys have been wrapped. Key wrapping is performed using AES KW, as specified in FCS_COP.1(d).

FDEAAcPP20E:FPT_PWR_EXT.1:

The Pre-Boot Authentication (PBA) portion of the TOE represents software executing on host computer and as such, supports only a single Compliant power saving state:

a) G3. In this state, the computing system is completely off and it does not consume any power. The system returns to the working state only after a complete reboot and hence PBA will be invoked/executed for authentication/authorization.

FDEEEcPP20E:FPT_PWR_EXT.1:

While the Self-Encrypting Drive (SED) (the hardware portion of the TOE) supports a single Compliant power state of device full off (D3, also named "D3cold" or "D3 (Off)"), the overall TOE supports only a the G3 state. Once the TOEs SED has been integrated into the computing platform controlled by Cigent's PBA software, the overall TOE only supports the G3 (off) power state.

FDEAAcPP20E:FPT_PWR_EXT.2:

The TOE enters a Compliant power saving state as prompted by the protected OS and user-initiated requests.

FDEEEcPP20E:FPT_PWR_EXT.2:

"A user-initiated request" is removing the power in the context of a SED.

Separately, the drive can be locked, but remains in a power on state. The requirement does not apply in this case.

FDEEEcPP20E:FPT_RBP_EXT.1:

The TOE supports the functional capability to assure that downgrading to a lower security version number is not possible. With this mechanism if a flaw in FW 1 is found then FW 2 is generated and downloaded to the drive. Using the internal block point mechanism, FW 1 will no longer be compatible with the drive and cannot be downloaded.

If a firmware update package is downloaded to the drive with an invalid firmware revision number, the RollBack protection firmware in the TOE generates and returns an error code and the firmware update package is rejected with one of the following error codes.

Roll back Error Messag

Error Stat Message

StatusType = 1h; StatusCode = 07h "Invalid Firmware Image"

FDEAAcPP20E:FPT_TST_EXT.1:

The TOE's PBA runs the following Power on Self-Tests:

- Power on Self-Tests:
  - AES CBC Encrypt/Decrypt KATs,
  - SHA-256/384/512 KAT,
  - RSA sign/verify KAT,
  - HMAC-SHA-256/384/512 KAT,
  - DRBG KAT,
  - DRBG Health Tests.
- Conditional tests:
  - DRBG Health Tests,

- Software Upgrade verification: RSA Signature Verification

FDEEEcPP20E:FPT_TST_EXT.1:

The TOE's SED runs the following Power on Self-Tests:

- Power on Self-Tests:
  - Hardware SHA-256, SHA-512: Digest KAT performed
  - Hardware RSA: Verify KAT performed
  - Firmware HMAC-SHA-256: HMAC KAT performed
  - Firmware XTS-AES-256: Encrypt and Decrypt KATs performed
  - Firmware RSA: Verify KAT performed
  - Firmware 800-90 DRBG: DRBG KAT performed
  - Firmware 800-132 PBKDF: PBKDF KAT performed
  - Firmware Integrity Check: Signature Verification
  - Firmware SHA-512: SHA-512 KAT performed
  - Secure boot process
- Conditional tests:
  - Firmware Load Check: RSA PSS signature verification of new firmware image is done before it can be loaded.
  - Firmware 800-90 DRBG: Newly generated random number is compared to the previously generated random number. Test fails if they are equal.
  - Firmware 800-90 DRBG Entropy: Newly generated random number is compared to the previously generated random number. Test fails if they are equal.
  - Firmware 800-38F Key Wrap: AES Key Wrap and Unwrap KATs performed.

FDEAAcPP20E:FPT_TUD_EXT.1:

Update files are digitally signed (RSA 4096 using SHA-512 per FCS_COP.1(a)) by Cigent and verified by the TOE prior to installation. The TOE does not support automatic update credentials. Only authorized administrators may manually perform the update process.

FDEEEcPP20E:FPT_TUD_EXT.1:

The TOE's SED provides authorized users with the ability to query the current version of the TOE firmware, the ability to initiate the TOE firmware updates, and the ability to verify updates (prior to installing those updates) using the RSA digital signature algorithm (with a key size (modulus) of 4096 bits and using SHA-512) provided by Cigent.