Samsung Electronics Co., Ltd. Samsung Knox File Encryption (PP_APP_V1.3/MOD_FE_V1.0) Security Target

Version: 0.5
2019/12/06

Prepared for:

SAMSUNG

Samsung Electronics Co., Ltd.

416 Maetan-3dong, Yeongtong-gu, Suwon-si, Gyeonggi-do, 443-742 Korea

Prepared By:

Gossamer Laboratories

www.gossamersec.com

Table of Contents

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 consists of the Samsung Knox File Encryption 1.0 provided by Samsung Electronics Co., Ltd.. The TOE is being evaluated as a File Encryption application.

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)

Acronyms and Terminology

AA	Assurance Activity
CC	Common Criteria
CCEVS	Common Criteria Evaluation and Validation Scheme
EAR	Entropy Analysis Report
FEK	File Encryption Key
GUI	Graphical User Interface
KEK	Key Encryption Key
MKDD	Master Key DualDAR
PCL	Product Compliant List
PMKEK	Password-based Master Key Encryption Key
PP	Protection Profile
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SOF	Strength of Function
ST	Security Target
TEE	Trusted Execution Environment (TrustZone)
TOE	Target of Evaluation
U.S.	United States
VR	Validation Report

Conventions

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 the ST, iteration is 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.
    - 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 – Samsung Electronics Co., Ltd. Samsung Knox File Encryption (PP_APP_V1.3/MOD_FE_V1.0) Security Target

ST Version – Version 0.5

ST Date – 2019/12/06

1.2 TOE Reference

TOE Identification – Samsung Knox File Encryption 1.0

TOE Developer – Samsung Electronics Co., Ltd.

Evaluation Sponsor – Samsung Electronics Co., Ltd.

1.3 TOE Overview

The Target of Evaluation (TOE) is Samsung Knox File Encryption 1.0. The TOE is a service built into the Knox Workspace container that can provide an additional layer of file encryption when a Workspace container is created with File Encryption enabled. This is available on devices that launch with Android 9 and Knox 3.3 or higher.

1.4 TOE Description

The TOE is a software service built into Samsung Android 9 with Knox 3.3 to provide file encryption to a Knox Workspace container. Samsung Knox File Encryption is designed to provide a second encryption layer similar to and on top of the FBE layer for the entire device, specifically for the Knox Workspace container.

The Knox File Encryption service runs in the background and utilizes Samsung Android cryptographic modules to provide file encryption services for the Knox Workspace container. The service is designed to run without any user intervention as all files in the Knox Workspace container will be encrypted automatically.

The Master Key (MKDD) is protected by a Trusted App inside TrustZone by the user's password. Each encrypted file is protected by a uniquely generated FEK which is encrypted by the Master Key as a KEK. When the user locks the container (or the container becomes locked due to inactivity), the Master Key and all FEKs are wiped from memory to fully lock the Knox Workspace container.

The following table shows the model numbers of the mobile devices used during evaluation testing of Knox File Encryption 1.0 (the version is listed as "DualDAR"):

Device Name	Model Number	Chipset Vendor	CPU	Android Version	TEE OS	Knox Version	DualDAR Version
Galaxy S10e	SM-G970F	Samsung	Exynos 9820	9	TEEGRIS 3.1	3.3	1.0.3
Galaxy S10+	SM-G975U	Qualcomm	SM8150	9	QSEE 5.2	3.3	1.0.2

Table 1 - Evaluated Devices

In addition to the evaluated devices, the following device models are claimed as equivalent with a note about the differences between the evaluated device and the equivalent models.

Evaluated Device	CPU	Equivalent Devices	Differences
Galaxy S10e (Samsung)	Exynos 9820	Galaxy S10+ (Samsung) Galaxy S10 5G (Samsung) Galaxy S10 (Samsung) Galaxy Note10+ 5G (Samsung) Galaxy Note10+ (Samsung) Galaxy Note10 5G (Samsung)	• All devices have same software environments (Android, Kernel, TEE) • Note10 devices have Exynos 9825 CPU
Galaxy S10+ (Qualcomm)	SM8150	Galaxy S10 5G (Qualcomm) Galaxy S10 (Qualcomm) Galaxy S10e (Qualcomm) Galaxy Fold (Qualcomm) Galaxy Note10+ 5G (Qualcomm) Galaxy Note10+ (Qualcomm) Galaxy Note10 (Qualcomm) Galaxy Tab S6	• All devices have same software environments (Android, Kernel, TEE)

Table 2 - Equivalent Devices

1.4.1 TOE Architecture

The TOE is software built into Samsung Knox. The TOE is designed as a framework for providing file encryption to all files within a Knox Workspace container where it is enabled. The software is comprised of four major components: the DualDAR Service, the DualDAR Client, the DualDAR Driver and cryptographic modules. Management of the TOE is provided through normal device administration functions; the TOE does not provide any configuration or management capabilities itself but relies on the platform to provide both UI (such as for password entry or management and MDM control).

Administration is limited to enabling the File Encryption feature when a Knox Workspace container is created.

The components provide the following functions within the TOE:

- DualDAR Service: manages the implementation of the configuration and monitoring system status for the lock state
- DualDAR Client: handles access to the Master Key (unlock and wipe)
- DualDAR Driver: handles the encryption/decryption I/O of files with the Master Key unlocked by the DualDAR Client
- Cryptographic Modules: handle the cryptographic operations of the TOE (Samsung Kernel Cryptographic Module and Samsung SCrypto)

The TOE utilizes the Knox Workspace container authentication (using a password) to unlock the 256-bit Master Key. Once the Master Key is unlocked the DualDAR Driver can read an encrypted file to unlock the individual 256-bit FEK. The unlocked FEK is then used to decrypt the contents. When the Knox Workspace container is locked, all open files will be closed and all unlocked FEKs and the Master Key will be cleared from memory (this is handled by the DualDAR Service).

By default (and in this configuration), the DualDAR Driver utilizes the Samsung Kernel Cryptographic Module of the device for AES-CBC-256 to decrypt/encrypt the contents of the file. The FEK is encrypted with AES-GCM using the 256-bit Master Key. All keys are generated using platform-provided DRBG functions and are 256-bit.

The TOE does not provide or utilize any communications services, nor does the TOE transmit or receive data or keys from remote systems.

Samsung provides a SDK which can be used to integrate a third-party encryption library to be used by the DualDAR Service and Driver but this configuration is not included as part of this evaluation.

1.4.1.1 Physical Boundaries

The TOE is a software application running on a mobile device. The mobile device platform provides a host Operating System and a Trusted Execution Environment.

1.4.1.2 Logical Boundaries

This section summarizes the security functions provided by the Samsung Knox File Encryption:

- Cryptographic support
- User data protection
- Identification and authentication
- Security management
- Privacy
- Protection of the TSF
- Trusted path/channels

1.4.1.2.1 Cryptographic Support

The TOE runs as part of Samsung Android 9 with Knox 3.3 and higher and includes several cryptographic libraries for encryption/decryption/cryptographic hashing functions for securing file contents and TOE keys. The TOE relies on the platform cryptographic functions for random number generation.

1.4.1.2.2 User Data Protection

The TOE protects all user data within the Knox Workspace container by providing an automatic encryption service for all files stored within the container; applications to not have to be made aware of the Knox File Encryption service to be protected. All keys are AES 256-bit, using AES-GCM for FEK protection and AES-CBC for file content protection.

1.4.1.2.3 Identification and Authentication

The TOE utilizes the authentication services provided by the Knox Workspace container to unlock the Master Key. Unsuccessful authentication to the Knox Workspace container will prevent the Master Key from being unlocked, and hence no files in the container can be accessed.

1.4.1.2.4 Security Management

The services provided by the TOE are not available until a Knox Workspace container with File Encryption enabled is created on the device. Authentication management and the container lock settings are handled by the Knox Workspace management and are generic for all Knox Workspace configurations.

1.4.1.2.5 Privacy

The TOE does not transmit Personally Identifiable Information over any network interfaces nor does it request access to any applications that may contain such information.

1.4.1.2.6 Protection of the TSF

The TOE relies on the physical boundary of the evaluated platform as well as the Samsung Android operating system for the protection of the TOE's components.

The TOE relies on the Samsung Android operating system to provide updates as the software is incorporated as part of the device image. The version of the Knox File Encryption software can be seen in the *About Device* page with the *Knox version* information (as the DualDAR version).

The TOE is a Samsung component, and all code is maintained solely by Samsung. Only documented APIs available in Samsung Android (which includes the Knox Workspace and Samsung cryptographic libraries) are used.

1.4.1.2.7 Trusted Path/Channels

The TOE does not transmit Personally Identifiable Information over any network interfaces.

1.4.2 TOE Documentation

- Samsung Knox File Encryption 1.0 Administrator Guide, version 1.0, October 8, 2019

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 Extended
- PP-Configuration for PP-Configuration for Application Software and File Encryption, Version 1.0, 25 July 2019 (CFG_APP-FE_V1.0)
  - The PP-Configuration includes the following components:
    - Base-PP: Protection Profile for Application Software, Version 1.3 (PP_APP_V1.3)
    - PP-Module: PP-Module for File Encryption, Version 1.0 (MOD_FE_V1.0)
- Technical Decisions as of December 5, 2019:

TD No.	PP	Applied	Rationale
0416	PP_APP_V1.3	Yes	
0427	PP_APP_V1.3	Yes	
0434	PP_APP_V1.3	No	Not a Windows Desktop application
0435	PP_APP_V1.3	No	Not a Linux system
0437	PP_APP_V1.3	Yes	
0444	PP_APP_V1.3	No	Not a VPN Client
0445	PP_APP_V1.3	Yes	
0455	MOD_FE_1.0	Yes	
0471	PP_APP_V1.3	Yes	
0472	MOD_FE_V1.0	Yes	

Table 3 - Technical Decisions

2.1 Conformance Rationale

The ST conforms to the CFG_APP-FE_V1.0. As explained previously, the security problem definition, security objectives, and security requirements are defined in the PP_APP_V1.3/MOD_FE_V1.0 that make up the CFG_APP-FE_V1.0.

3 Security Objectives

The Security Problem Definition may be found in the PP_APP_V1.3/MOD_FE_V1.0 and this section reproduces only the corresponding Security Objectives for operational environment for reader convenience. The PP_APP_V1.3/MOD_FE_V1.0 offers additional information about the identified security objectives, but that has not been reproduced here and the PP_APP_V1.3/MOD_FE_V1.0 should be consulted if there is interest in that material.

In general, the PP_APP_V1.3/MOD_FE_V1.0 has defined Security Objectives appropriate for Mobile Devices and as such are applicable to the Samsung Knox File Encryption TOE.

3.1 Security Objectives for the Operational Environment

- **OE.AUTHORIZATION_FACTOR_STRENGTH** (MOD_FE_V1.0) An authorized user will be responsible for ensuring that all externally derived authorization factors have sufficient strength and entropy to reflect the sensitivity of the data being protected. This can apply to password or passphrase based, ECC CDH, and RSA authorization factors.
- **OE.PLATFORM** (PP_APP_V1.3) The TOE relies upon a trustworthy computing platform for its execution. This includes the underlying operating system and any discrete execution environment provided to the TOE.
- **OE.POWER_SAVE** (MOD_FE_V1.0) The non-mobile operational environment must be configurable so that there exists at least one mechanism that will cause the system to enter a safe power state (A.SHUTDOWN). Any such mechanism (e.g., sleep, hibernate) that does not conform to this requirement must be capable of being disabled. The mobile operational environment must be configurable such that there exists at least one mechanism that will cause the system to lock upon a period of time.
- **OE.PROPER_USER** (PP_APP_V1.3) The user of the application software is not willfully negligent or hostile, and uses the software within compliance of the applied enterprise security policy.
- **OE.PROPER_ADMIN** (PP_APP_V1.3) The administrator of the application software is not careless, willfully negligent or hostile, and administers the software within compliance of the applied enterprise security policy.
- **OE.STRONG_ENVIRONMENT_ CRYPTO** (MOD_FE_V1.0) The Operating environment will provide a cryptographic function capability that is commensurate with the requirements and capabilities of the TOE.

4 Extended Components Definition

All of the extended requirements in this ST have been drawn from the PP_APP_V1.3/MOD_FE_V1.0. The PP_APP_V1.3/MOD_FE_V1.0 defines the following extended SFRs and SARs and since they are not redefined in this ST the PP_APP_V1.3/MOD_FE_V1.0 should be consulted for more information concerning those CC extensions.

Requirement Class	Requirement Component
FCS: Cryptographic support	MOD_FE_V1.0: FCS_CKM_EXT.2: Cryptographic key generation (FEK)
	PP_APP_V1.3: FCS_CKM_EXT.1(1): Cryptographic Key Generation Services
	MOD_FE_V1.0: FCS_CKM_EXT.2: Cryptographic key generation (FEK)
	MOD_FE_V1.0: FCS_CKM_EXT.3: Key Encrypting Key (KEK) Support
	MOD_FE_V1.0: FCS_CKM_EXT.4: Cryptographic Key Destruction
	MOD_FE_V1.0: FCS_CKM_EXT.6: Cryptographic Password/Passphrase Conditioning
	MOD_FE_V1.0: FCS_IV_EXT.1: Initialization Vector Generation
	MOD_FE_V1.0: FCS_KDF_EXT.1 Cryptographic Key Derivation Function
	MOD_FE_V1.0: FCS_KYC_EXT.1: Key Chaining and Key Storage
	PP_APP_V1.3: FCS_RBG_EXT.1: Random Bit Generation Services
	PP_APP_V1.3: FCS_STO_EXT.1: Storage of Credentials
	MOD_FE_V1.0: FCS_VAL_EXT.1: Validation
FDP: User data protection	PP_APP_V1.3: FDP_DAR_EXT.1: Encryption of Sensitive Application Data
	PP_APP_V1.3: FDP_DEC_EXT.1: Access to Platform Resources
	PP_APP_V1.3: FDP_NET_EXT.1: Network Communications
	MOD_FE_V1.0: FDP_PM_EXT.1: Protection of Data in Power Managed States
	MOD_FE_V1.0: FDP_PRT_EXT.1: Protection of Selected User Data
	MOD_FE_V1.0: FDP_PRT_EXT.2: Destruction of Plaintext Data
	MOD_FE_V1.0: FDP_PRT_EXT.3: Protection of Third-Party Data
FIA: Identification and authentication	MOD_FE_V1.0: FIA_AUT_EXT.1: User Authorization
FMT: Security management	PP_APP_V1.3: FMT_CFG_EXT.1: Secure by Default Configuration
	PP_APP_V1.3: FMT_MEC_EXT.1: Supported Configuration Mechanism
FPR: Privacy	PP_APP_V1.3: FPR_ANO_EXT.1: User Consent for Transmission of Personally Identifiable Information
FPT: Protection of the TSF	PP_APP_V1.3: FPT_AEX_EXT.1: Anti-Exploitation Capabilities
	PP_APP_V1.3: FPT_API_EXT.1: Use of Supported Services and APIs
	PP_APP_V1.3: FPT_IDV_EXT.1: Software Identification and Versions
	MOD_FE_V1.0: FPT_KYP_EXT.1: Protection of Key and Key Material
	PP_APP_V1.3: FPT_LIB_EXT.1: Use of Third Party Libraries
	PP_APP_V1.3: FPT_TUD_EXT.1: Integrity for Installation and Update
FTP: Trusted Path/Channels	PP_APP_V1.3: FTP_DIT_EXT.1: Protection of Data in Transit
ALC: Life Cycle Support	PP_APP_V1.3: ALC_TSU_EXT.1: Timely Security Updates

Table 4 - Extended SFRs and SARs

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 PP_APP_V1.3/MOD_FE_V1.0. The refinements and operations already performed in the PP_APP_V1.3/MOD_FE_V1.0 are not identified (e.g., highlighted) here, rather the requirements have been copied from the PP_APP_V1.3/MOD_FE_V1.0 and any residual operations have been completed herein. Of particular note, the PP_APP_V1.3/MOD_FE_V1.0 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 PP_APP_V1.3/MOD_FE_V1.0, which include all the SARs for EAL 1 augmented with ALC_TSU_EXT.1. However, the SARs are effectively refined since requirement-specific 'Assurance Activities' are defined in the PP_APP_V1.3/MOD_FE_V1.0 that serve to ensure corresponding evaluations will yield more practical and consistent assurance than the assurance requirements alone. The PP_APP_V1.3/MOD_FE_V1.0 should be consulted for the assurance activity definitions.

5.1 TOE Security Functional Requirements

The following table identifies the SFRs that are satisfied by the Samsung Knox File Encryption TOE.

Requirement Class	Requirement Component
FCS: Cryptographic support	PP_APP_V1.3: FCS_CKM.1(2): Cryptographic Symmetric Key Generation
	PP_APP_V1.3: FCS_CKM.1(3): Password Conditioning
	PP_APP_V1.3: FCS_CKM_EXT.1(1): Cryptographic Key Generation Services
	MOD_FE_V1.0: FCS_CKM_EXT.2: Cryptographic key generation (FEK)
	MOD_FE_V1.0: FCS_CKM_EXT.3: Key Encrypting Key (KEK) Support
	MOD_FE_V1.0: FCS_CKM_EXT.4: Cryptographic Key Destruction
	MOD_FE_V1.0: FCS_CKM_EXT.6: Cryptographic Password/Passphrase Conditioning
	PP_APP_V1.3: FCS_COP.1(1): Cryptographic Operation – Encryption/Decryption
	PP_APP_V1.3: FCS_COP.1(4): Cryptographic Operation - Keyed-Hash Message Authentication
	MOD_FE_V1.0: FCS_COP.1(5): Cryptographic Operation (Key Wrapping)
	MOD_FE_V1.0: FCS_IV_EXT.1: Initialization Vector Generation
	MOD_FE_V1.0: FCS_KYC_EXT.1: Key Chaining and Key Storage
	PP_APP_V1.3: FCS_RBG_EXT.1: Random Bit Generation Services
	PP_APP_V1.3: FCS_STO_EXT.1: Storage of Credentials
	MOD_FE_V1.0: FCS_VAL_EXT.1: Validation
FDP: User data protection	PP_APP_V1.3: FDP_DAR_EXT.1: Encryption of Sensitive Application Data
	PP_APP_V1.3: FDP_DEC_EXT.1: Access to Platform Resources
	PP_APP_V1.3: FDP_NET_EXT.1: Network Communications

Requirement Class	Requirement Component
	MOD_FE_V1.0: FDP_PM_EXT.1: Protection of Data in Power Managed States
	MOD_FE_V1.0: FDP_PRT_EXT.1: Protection of Selected User Data
	MOD_FE_V1.0: FDP_PRT_EXT.2: Destruction of Plaintext Data
	MOD_FE_V1.0: FDP_PRT_EXT.3: Protection of Third-Party Data
FIA: Identification and authentication	MOD_FE_V1.0: FIA_AUT_EXT.1: User Authorization
FMT: Security management	PP_APP_V1.3: FMT_CFG_EXT.1: Secure by Default Configuration
	PP_APP_V1.3: FMT_MEC_EXT.1: Supported Configuration Mechanism
	PP_APP_V1.3: FMT_SMF.1: Specification of Management Functions
	MOD_FE_V1.0: FMT_SMF.1(2): Specification of Management Functions
FPR: Privacy	PP_APP_V1.3: FPR_ANO_EXT.1: User Consent for Transmission of Personally Identifiable Information
FPT: Protection of the TSF	PP_APP_V1.3: FPT_AEX_EXT.1: Anti-Exploitation Capabilities
	PP_APP_V1.3: FPT_API_EXT.1: Use of Supported Services and APIs
	PP_APP_V1.3: FPT_IDV_EXT.1: Software Identification and Versions
	MOD_FE_V1.0: FPT_KYP_EXT.1: Protection of Key and Key Material
	PP_APP_V1.3: FPT_LIB_EXT.1: Use of Third Party Libraries
	PP_APP_V1.3: FPT_TUD_EXT.1: Integrity for Installation and Update
FTP: Trusted Path/Channels	PP_APP_V1.3: FTP_DIT_EXT.1: Protection of Data in Transit

Table 5 – TOE Security Functional Requirements

5.1.1 Cryptographic Support (FCS)

5.1.1.1 PP_APP_V1.3: FCS_CKM.1(2): Cryptographic Symmetric Key Generation

FCS_CKM.1(2).1

The **application** shall generate **symmetric** cryptographic keys **using a Random Bit Generator as specified in FCS_RBG_EXT.1** and specified cryptographic key sizes [

- ***256 bit***

].

5.1.1.2 PP_APP_V1.3: FCS_CKM.1(3): Password Conditioning

FCS_CKM.1(3).1

Refinement: A password/passphrase shall perform [Password-based Key Derivation Functions] in accordance with a specified cryptographic algorithm as specified in FCS_COP.1(4), with [**100,000**] iterations, and output cryptographic key sizes [***256***] that meet the following **[NIST SP 800-132]**.

FCS_CKM.1(3).2

The TSF shall generate salts using a RBG that meets FCS_RGB_EXT.1 and with entropy corresponding to the security strength selected for PBKDF in FCS_CKM.1(3).1.

5.1.1.3 PP_APP_V1.3: FCS_CKM_EXT.1(1): Cryptographic Key Generation Services

FCS_CKM_EXT.1(1).1

The application shall [

- ***generate no asymmetric cryptographic keys***

].

5.1.1.4 MOD_FE_V1.0: FCS_CKM_EXT.2: Cryptographic key generation (FEK)

FCS_CKM_EXT.2.1

The TSF shall [

- ***generate FEK cryptographic keys [***
  - ***using a Random Bit Generator as specified in FCS_RBG_EXT.1 (from [AppPP]) and with entropy corresponding to the security strength of AES key sizes of [256 bit]***

  ]

].

FCS_CKM_EXT.2.2

The TSF shall use a unique FEK for each file (or set of files) using the mechanism on the client as specified in FCS_CKM_EXT.2.1.

5.1.1.5 MOD_FE_V1.0: FCS_CKM_EXT.3: Key Encrypted Key (KEK) Support

FCS_CKM_EXT.3.1

The TSF shall [

- ***generate KEK cryptographic keys [***
  - ***using a Random Bit Generator as specified in FCS_RBG_EXT.1 (from [AppPP]) and with entropy corresponding to the security strength of AES key sizes of [256 bit]***
  - ***derived from a password/passphrase that is conditioned as defined in FCS_CKM_EXT.6]***

].

5.1.1.6 MOD_FE_V1.0: FCS_CKM_EXT.4: Cryptographic Key Destruction

FCS_CKM_EXT.4.1

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, the destruction shall be executed by [***
  - ***the invocation of an interface provided by the underlying platform that [***
    - ***instructs the underlying platform to destroy the abstraction that represents the key]***

].

FCS_CKM_EXT.4.2

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

5.1.1.7 MOD_FE_V1.0: FCS_CKM_EXT.6: Cryptographic Password/Passphrase Conditioning

FCS_CKM_EXT.6.1

The TSF shall support a password/passphrase of up to [***maximum value supported by the platform: 16***] characters used to generate a password authorization factor.

FCS_CKM_EXT.6.2

The TSF shall allow passwords to be composed of any combination of upper case characters, lower case characters, numbers, and the following special characters: "!", "@", "#", "$", "%", "^", "&", "*", "(", and ")", and [***+ = _ / - ' " : ; , ? ` ~ \ | < > { } [ ]***].

FCS_CKM_EXT.6.3

The TSF shall perform Password-based Key Derivation Functions in accordance with a specified cryptographic algorithm [**HMAC-**[***SHA-256***]], with [**100,000**] iterations, and output cryptographic key sizes [***256***] that meet the following: [*NIST SP 800-132*].

FCS_CKM_EXT.6.4

The TSF shall not accept passwords less than [***4 characters***] and greater than the maximum password length defined in FCS_CKM_EXT.6.1.

FCS_CKM_EXT.6.5

The TSF shall generate all salts using an RBG that meets FCS_RBG_EXT.1 (**from [AppPP]**) and with entropy corresponding to the security strength selected for PBKDF in FCS_CKM_EXT.6.3

5.1.1.8 PP_APP_V1.3: FCS_COP.1(1): Cryptographic Operation – Encryption/Decryption

FCS_COP.1(1).1

The **application** shall perform *encryption/decryption* in accordance with a specified cryptographic algorithm [

- ***AES-CBC (as defined in NIST SP 800-38A) mode,***
- ***AES-GCM (as defined in NIST SP 800-38D) mode,***

] and cryptographic key sizes [

- ***256 bit***

].

5.1.1.9 PP_APP_V1.3: FCS_COP.1(4): Cryptographic Operation - Keyed-Hash Message Authentication

FCS_COP.1(4).1

The **application** shall perform *keyed-hash message authentication* in accordance with a specified cryptographic algorithm

- HMAC-SHA-256

and [

- ***no other algorithms***

] with key sizes [**256-bits**] and message digest sizes 256 and [***no other size***] bits that meet the following: FIPS Pub 198-1 *The Keyed-Hash Message Authentication Code* and FIPS Pub 180-4 *Secure Hash Standard*.

5.1.1.10 MOD_FE_V1.0: FCS_COP.1(5): Cryptographic Operation (Key Wrapping)

FCS_COP.1(5).1

The TSF shall [***implement functionality to perform Key Wrapping***] in accordance with a specified cryptographic algorithm [*AES*] **in the following modes** [

- ***GCM mode***

] and the cryptographic key size [***256 bits (AES)***] that meet the following: [

- ***"NIST SP 800-38D"***

] **and no other standards**.

5.1.1.11 MOD_FE_V1.0: FCS_IV_EXT.1: Initialization Vector Generation

FCS_IV_EXT.1.1

The TSF shall [

- ***implement platform-provided functionality to generate IVs***

].

5.1.1.12 MOD_FE_V1.0: FCS_KDF_EXT.1: Cryptographic Key Derivation Function

FCS_KDF_EXT.1.1

The TSF shall [*accept* [***a conditioned password***] to derive an intermediate key, as defined in [

- ***NIST SP 800-132***

*] using the keyed-hash functions specified in FCS_COP.1(4) (**from [AppPP]**), such that the output is at least of equivalent security strength (in number of bits) to the [FEK]*].

5.1.1.13 MOD_FE_V1.0: FCS_KYC_EXT.1: Key Chaining and Key Storage

FCS_KYC_EXT.1.1

The TSF shall maintain a key chain of [

- ***[KEKs] originating from [one or more authorization factors(s)] to [the FEK(s)] using the following method(s): [***
  - ***implementation of key wrapping as specified in FCS_COP.1(5),***
  - ***implementation of key derivation as specified in FCS_KDF_EXT.1***

  ] while maintaining an effective strength of [
  - ***[256 bits] for symmetric keys;***

  ] commensurate with the strength of the FEK]

and [

- ***no supplemental key chains]***

].

5.1.1.14 PP_APP_V1.3: FCS_RBG_EXT.1: Random Bit Generation Services

FCS_RBG_EXT.1.1

The application shall [

- ***invoke platform-provided DRBG functionality***

] for its cryptographic operations.

5.1.1.15 PP_APP_V1.3: FCS_STO_EXT.1: Storage of Credentials

FCS_STO_EXT.1.1

The application shall [

- ***implement functionality to securely store [MKDD and FEKs] according to [FCS_COP.1(1)]***
- ***implement functionality to securely store [PMKEK] according to [FCS_CKM.1(3)]***

] to non-volatile memory.

5.1.1.16 MOD_FE_V1.0: FCS_VAL_EXT.1 Validation

FCS_VAL_EXT.1.1

The TSF shall perform validation of the [*user*] by [

- ***receiving assertion of the subject's validity from [Knox Workspace container]***

].

FCS_VAL_EXT.1.2

The TSF shall require validation of the [*user*] prior to [*decrypting any FEK*].

5.1.2 User Data Protection (FDP)

5.1.2.1 PP_APP_V1.3: FDP_DAR_EXT.1: Encryption Of Sensitive Application Data

FDP_DAR_EXT.1.1

The application shall [

- ***implement functionality to encrypt sensitive data as defined in the EP for File Encryption;***
- ***protect sensitive data in accordance with FCS_STO_EXT.1***

].

5.1.2.2 PP_APP_V1.3: FDP_DEC_EXT.1: Access to Platform Resources

FDP_DEC_EXT.1.1

The application shall restrict its access to [

- ***no hardware resources***

].

FDP_DEC_EXT.1.2

The application shall restrict its access to [

- ***no sensitive information repositories***

].

5.1.2.3 PP_APP_V1.3: FDP_NET_EXT.1: Network Communications

FDP_NET_EXT.1.1

The application shall restrict network communication to [

- ***no network communications***

].

5.1.2.4 MOD_FE_V1.0: FDP_PM_EXT.1: Protection of Data in Power Managed States

FDP_PM_EXT.1.1

The TSF shall protect all data selected for encryption during the transition to the [**Data Locked**] state as per FDP_PRT_EXT.1.1.

FDP_PM_EXT.1.2

On the return to a powered-on state from the state(s) indicated in FDP_PM_EXT.1.1, the TSF shall authorize the user in the manner specified in FIA_AUT_EXT.1.1 once before any protected data are decrypted.

FDP_PM_EXT.1.3

The TSF shall destroy all key material and authentication factors stored in plaintext when transitioning to a protected state as defined by FDP_PM_EXT.1.1.

5.1.2.5 MOD_FE_V1.0: FDP_PRT_EXT.1: Protection of Selected User Data

FDP_PRT_EXT.1.1

The TSF shall perform encryption and decryption of the user-selected file (or set of files) in accordance with FCS_COP.1(1) (**from [AppPP]**).

FDP_PRT_EXT.1.2

The TSF shall [***implement functionality***] to ensure that all sensitive data created by the TOE when decrypting/encrypting the user-selected file (or set of files) are destroyed in volatile and non-volatile memory when the data is no longer needed according to FCS_CKM_EXT.4.

5.1.2.6 MOD_FE_V1.0: FDP_PRT_EXT.2: Destruction of Plaintext Data

FDP_PRT_EXT.2.1

The TSF shall [***implement functionality***] to ensure that all original plaintext data created when decrypting/encrypting the user-selected file (or set of files) are destroyed in volatile and non-volatile memory according to FCS_CKM_EXT.4 upon completion of the decryption/encryption operation.

5.1.2.7 MOD_FE_V1.0: FDP_PRT_EXT.3: Protection of Third-Party Data

FDP_PRT_EXT.3.1

The TSF shall ensure that all temporary files created by [***all applications***] when decrypting/encrypting the user-selected file (or set of files) are removed or encrypted upon completion of the decryption/encryption operation.

5.1.3 Identification and Authentication (FIA)

5.1.3.1 MOD_FE_V1.0: FIA_AUT_EXT.1: User Authorization

FIA_AUT_EXT.1.1

The application shall [***implement platform-provided functionality to provide user authorization***] based on [

- ***a password authorization factor conditioned as defined in FCS_CKM_EXT.6***

].

5.1.4 Security Management (FMT)

5.1.4.1 PP_APP_V1.3: FMT_CFG_EXT.1: Secure by Default Configuration

FMT_CFG_EXT.1.1

The application shall provide only enough functionality to set new credentials when configured with default credentials or no credentials.

FMT_CFG_EXT.1.2

The application shall be configured by default with file permissions which protect the application binaries and data files from modification by normal unprivileged users.

5.1.4.2 PP_APP_V1.3: FMT_MEC_EXT.1: Supported Configuration Mechanism

FMT_MEC_EXT.1.1

The TSF shall [***invoke the mechanisms recommended by the platform vendor for storing and setting configuration options***].

(TD0437 applied)

5.1.4.3 PP_APP_V1.3: FMT_SMF.1: Specification of Management Functions

FMT_SMF.1.1

The TSF shall be capable of performing the following management functions [

- ***specify a time period after device lock to enter the Data Lock state***

].

5.1.4.4 MOD_FE_V1.0: FMT_SMF.1(2): Specification of Management Functions

FMT_SMF.1(2).1

The TSF shall be capable of performing the following management functions: [

- ***perform a cryptographic erase of the data by the destruction of FEKs or KEKs protecting the FEKs as described in FCS_CKM_EXT.4.1,***

].

5.1.5 Privacy (FPR)

5.1.5.1 PP_APP_V1.3: FPR_ANO_EXT.1: User Consent for Transmission of Personally Identifiable Information

FPR_ANO_EXT.1.1

The application shall [

- ***not transmit PII over a network***

].

5.1.6 Protection of the TSF (FPT)

5.1.6.1 PP_APP_V1.3: FPT_AEX_EXT.1: Anti-Exploitation Capabilities

FPT_AEX_EXT.1.1

The application shall not request to map memory at an explicit address except for [**no exceptions**].

FPT_AEX_EXT.1.2

The application shall [

- ***not allocate any memory region with both write and execute permissions***

].

FPT_AEX_EXT.1.3

The application shall be compatible with security features provided by the platform vendor.

FPT_AEX_EXT.1.4

The application shall not write user-modifiable files to directories that contain executable files unless explicitly directed by the user to do so.

FPT_AEX_EXT.1.5

The application shall be built with stack-based buffer overflow protection enabled.

5.1.6.2 PP_APP_V1.3: FPT_API_EXT.1: Use of Supported Services and APIs

FPT_API_EXT.1.1

The application shall use only documented platform APIs.

5.1.6.3 PP_APP_V1.3: FPT_IDV_EXT.1: Software Identification and Versions

FPT_IDV_EXT.1.1

The application shall be versioned with [***[an identifier in the Security software version display]***].

5.1.6.4 MOD_FE_V1.0: FPT_KYP_EXT.1: Protection of Key and Key Material

FPT_KYP_EXT.1.1

The TSF shall [

- ***store keys in non-volatile memory only when [***
  - ***wrapped, as specified in FCS_COP.1(5)]***

].

5.1.6.5 PP_APP_V1.3: FPT_LIB_EXT.1: Use of Third Party Libraries

FPT_LIB_EXT.1.1

The application shall be packaged with only [**no third party libraries**].

5.1.6.6 PP_APP_V1.3: FPT_TUD_EXT.1: Integrity for Installation and Update

FPT_TUD_EXT.1.1

The application shall [***leverage the platform***] to check for updates and patches to the application software.

FPT_TUD_EXT.1.2

The application shall [***leverage the platform***] to query the current version of the application software.

FPT_TUD_EXT.1.3

The application shall not download, modify, replace or update its own binary code.

FPT_TUD_EXT.1.4

The application installation package and its updates shall be digitally signed such that its platform can cryptographically verify them prior to installation.

FPT_TUD_EXT.1.5

The application is distributed [***with the platform OS***].

5.1.7 Trusted Path/Channels (FTP)

5.1.7.1 PP_APP_V1.3: FTP_DIT_EXT.1: Protection of Data in Transit

FTP_DIT_EXT.1.1

The application shall [

- ***not transmit any [data]***

] between itself and another trusted IT product.

5.2 TOE Security Assurance Requirements

The SARs for the TOE are the EAL 1 augmented with ALC_TSU_EXT.1 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.

5.2.1 Development (ADV)

5.2.1.1 ADV_FSP.1: Basic Functional Specification

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 AGD_OPE.1: Operational User Guidance

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 AGD_PRE.1: Preparative Procedures

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 ALC_CMC.1: Labelling of the TOE

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 ALC_CMS.1: TOE CM Coverage

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.

5.2.3.3 PP_APP_V1.3: ALC_TSU_EXT.1: Timely Security Updates

ALC_TSU_EXT.1.1d

The developer shall provide a description in the TSS of how timely security updates are made to the TOE.

ALC_TSU_EXT.1.1c

The description shall include the process for creating and deploying security updates for the TOE software.

ALC_TSU_EXT.1.2c

The description shall express the time window as the length of time, in days, between public disclosure of a vulnerability and the public availability of security updates to the TOE.

ALC_TSU_EXT.1.3c

The description shall include the mechanisms publicly available for reporting security issues pertaining to the TOE.

ALC_TSU_EXT.1.4c

The description shall include where users can seek information about the availability of new updates including details (e.g. CVE identifiers) of the specific public vulnerabilities corrected by each update.

ALC_TSU_EXT.1.1e

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

5.2.4 Tests (ATE)

5.2.4.1 ATE_IND.1: Independent Testing - sample

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.

5.2.5 Vulnerability Assessment (AVA)

5.2.5.1 AVA_VAN.1: Vulnerability Survey

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
- Identification and authentication
- Security management
- Privacy
- Protection of the TSF
- Trusted path/channels

6.1 Cryptographic Support

PP_APP_V1.3: FCS_CKM.1(2)
The TOE generates 256-bit symmetric keys using platform-provided cryptographic modules as described by FCS_RBG_EXT.1. The platform entropy pools are seeded continually from the hardware noise source to ensure sufficient entropy is available for the generation of keys.

PP_APP_V1.3: FCS_CKM.1(3)/MOD_FE_V1.0: FCS_KDF_EXT.1
The TOE conditions the user password using a PBKDF2 (using HMAC-SHA-256) function that meets NIST SP 800-132 with 100,000 rounds to generate a 256-bit KEK to protect the Master Key (MKDD). All salts are generated using RBGs as specified in FCS_RBG_EXT.1.

PP_APP_V1.3: FCS_CKM_EXT.1(1)
The TOE does not generate or use asymmetric keys.

MOD_FE_V1.0: FCS_CKM_EXT.2
The TOE generates FEKs utilizing the Samsung Linux Kernel that is part of the platform. The Kernel provides random strings, seeded from /dev/urandom with sufficient entropy to generate keys with 256-bit security strength.

The TOE automatically generates a new FEK every time a new file is created (as all files within the Workspace container will be encrypted). The FEK is stored (encrypted by the MKDD with AES-GCM) as part of a metadata header attached to the file. No user interaction is required to generate the FEK or to encrypt any file within the Workspace container.

MOD_FE_V1.0: FCS_CKM_EXT.3
The TOE generates two KEKs. The first KEK is the PMKEK and the second KEK is the MKDD. The PMKEK is derived, via PBKDF2 (HMAC-SHA-256), from the password entered by the user at successful authentication to the Knox Workspace container. The MKDD is generated by a platform DRBG seeded by raw output from the hardware noise source (similar to /dev/random inside the TEE) with sufficient entropy to generate keys with 256-bit strength. The key is then encrypted by the PMKEK to be stored in non-volatile memory. Both KEKs are 256-bit.

MOD_FE_V1.0: FCS_CKM_EXT.4
The TOE relies on the platform for destroying cryptographic keys when they are no longer in use. The platform cryptographic modules provide functionality to automatically overwrite (with zeros) of keys

when they are released. For keys stored in Flash the TOE calls the platform to perform a block erase to the flash controller. The TOE only writes encrypted keys to flash storage

MOD_FE_V1.0: FCS_CKM_EXT.6
The TOE utilizes the platform authentication services of the Knox Workspace container to access the user password. The platform supports passwords between 4-16 characters in length. The platform administrator may choose to require any password length within this range. The platform authenticates the user through a password consisting of basic Latin characters (upper and lower case, numbers, and the special characters noted in the selection (see section 5.1.1.7)).

The entered password is conditioned with 100,000 rounds of PBKDF2 (using HMAC-SHA-256) to generate a 256-bit KEK. This KEK is used to encrypt the 256-bit MKDD using AES-GCM.

PP_APP_V1.3: FCS_COP.1(1)/(4)/(5)
The TOE performs cryptographic algorithms in accordance with the following NIST standards and has received the following CAVP algorithm certificates.

The Samsung Kernel Cryptographic ("Kernel Crypto") Module provides the following algorithms used by the TOE.

Algorithm	NIST Standard	SFR Reference	Cert#
AES 256 CBC	FIPS 197, SP 800-38A	FCS_COP.1(1)	C:879, C:860, C:406, C:354
HMAC SHA-256	FIPS 198-1 & 180-4	FCS_COP.1(4)	C:879, C:860, C:406, C:354
SHS SHA-256	FIPS 180-4	FCS_COP.1(2)	C:879, C:860, C:406, C:354

Table 6 - Samsung Kernel Cryptographic Algorithms

The evaluated devices utilize the Samsung SCrypto Cryptographic Module for cryptographic operations within the TEE on each device. The following table lists the TEE operating systems for each device.

SoC	TEE OS Version	SCrypto Version
Samsung Exynos 9825/9820	TEEGRIS 3.1	2.4
Qualcomm SM8150	QSEE 5.0	2.4

Table 7 - TEE Environments

The Samsung SCrypto TEE library provides the following algorithms.

Algorithm	NIST Standard	SFR Reference	Cert#
AES CBC/GCM 256	FIPS 197, SP 800-38A/D	FCS_COP.1(1)	C:428

Table 8 - SCrypto TEE Cryptographic Algorithms

The TOE invokes The Samsung Kernel Cryptographic and the SCrypto modules to perform AES-GCM encryption of KEKs and FEKs. The file contents are encrypted using AES-CBC from the Samsung Kernel Cryptographic module. All keys used by the TOE are 256-bit. The key sizes and algorithms used cannot be changed.

MOD_FE_V1.0: FCS_IV_EXT.1
The TOE generates IVs for data storage encryption in for AES-CBC. The IVs are generated by /dev/urandom from the Android Linux kernel when the AES key for a file is created.

MOD_FE_V1.0: FCS_KYC_EXT.1
The PMKEK, MKDD and all FEKs are 256-bit AES keys. The MKDD is protected by the PMKEK, which is derived from the Knox Workspace container password using PBKDF2. The encrypted MKDD is stored in TrustZone as part of the DualDAR Client Trusted App data. The MKDD is used to encrypt the FEK of each file in the Knox Workspace container (the FEK is stored in the associated file metadata). All keys are encrypted with AES-GCM.

PP_APP_V1.3: FCS_RBG_EXT.1
The TOE utilizes a number of different RBGs included in the platform:

1. Raw entropy (equivalent to /dev/random direct from the hardware noise source) provided by the TEE OS
2. /dev/urandom from the Android Linux kernel

Each of these entropy sources will generate a 256-bit string. The APIs used by the TOE are available to applications that are part of the system services (not general Android applications).

PP_APP_V1.3: FCS_STO_EXT.1
The TOE uses a TrustZone application to protect the MKDD that is unlocked when the user successfully authenticates to the Knox Workspace container. The AES-GCM encrypted key is stored within the TrustZone boundary.

The TOE protects individual FEKs (and IVs) by encrypting them using AES-GCM with the MKDD and placing it in the file metadata.

The PMKEK is conditioned from the password provided by the Knox Workspace container using a PBKDF2 function following NIST SP 800-132.

MOD_FE_V1.0: FCS_VAL_EXT.1
The TOE relies on the platform to validate the user credentials via the authentication interface for the Knox Workspace container. A successful authentication will provide the TOE with access to the entered password to begin the process for accessing the MKDD and hence the protected files.

The MKDD can only be decrypted by the PMKEK, which must be conditioned from the user's authentication credentials. This ensures that the files cannot be accessed without a successful user authentication to the Knox Workspace container.

6.2 User Data Protection

PP_APP_V1.3: FDP_DAR_EXT.1
The TOE leverages the platform TrustZone for protection of the MKDD. The TOE includes a Trusted App which leverages an encryption module within TrustZone to encrypt (and decrypt) the MKDD.

To encrypt files within the Knox Workspace container, the TOE implements a service which will intercept file requests to encrypt (and decrypt) the files using the included cryptographic modules. All files written inside the Knox Workspace container boundary will be encrypted.

PP_APP_V1.3: FDP_DEC_EXT.1
The TOE is bundled as part of the application package for the Knox Workspace container. This package is bundled as part of the device image and is only updated as part of OS updates. The Knox Workspace container is installed automatically as a system service on the first boot. The TOE is only available when activated in a Knox Workspace container, and then only intercepts read/write requests for any files contained within the container boundary. The TOE does not access any sensitive information

repositories, but will encrypt/decrypt any application and data stored within the Knox Workspace container since it interacts at the file system level (below the concept of whether an application is sensitive or not).

PP_APP_V1.3: FDP_NET_EXT.1
The TOE does not transmit any information over a network.

MOD_FE_V1.0: FDP_PM_EXT.1
The TOE supports three discrete states on the device:

1. Off: the device is powered off (i.e. no power to the device)
2. Data Lock: the device is powered on, not authenticated to the TOE
3. Data Unlock: the device is powered on, authenticated to the TOE

The TOE spends most of its time between states two (Data Lock) and three (Data Unlock). It is not possible to move directly from Off to Data Unlock; the TOE always starts in the Data Lock state on startup.

During the Off state, the MKDD and any unlocked FEKS that may be in use are removed from volatile storage automatically, regardless of whether this state was by user shutdown/restart or by power failure.

On startup, the device places the TOE into the Data Lock state. From the Data Lock state, the user must authenticate successfully to the Knox Workspace container to transition to the Data Unlock state. How the TOE came to be in the Data Lock state (i.e. either directly from Off or from Data Unlock), does not change the functionality of the state; in either case successful authentication is required to move to the Data Unlock state. The authentication will allow the MKDD to be decrypted, which in turn will allow the FEKs to be decrypted.

When the TOE transitions to the Data Lock state from the Data Unlock state, the MKDD and any FEKs are cleared from volatile storage and any open files are closed. The transition to the Data Lock state can be initiated by the device being locked or by a timeout period (tied to inactivity in the Workspace container or by an administrator configuration). These settings are part of the Knox Workspace container configuration (the settings control when the container is locked) and not specifically TOE configurations, but are used to determine when the keys and files should be cleared from volatile storage.

MOD_FE_V1.0: FDP_PRT_EXT.1/ MOD_FE_V1.0: FDP_PRT_EXT.2
The TOE automatically encrypts all files within the Knox Workspace container, regardless of the point of creation (i.e. a direct user request to create a file vs. an application creating the file on behalf of the user). No temporary files are used during the encryption/decryption process as all file content access is handled through in-place buffers. The TOE only encrypts the file contents; file metadata is not encrypted (the encrypted version of the FEK is considered to be metadata).

The TOE is invoked when the Knox Workspace container is unlocked by the user entering their credentials. Until successful container authentication, all files are encrypted and not accessible. When the container enters the Data Lock state, the MKDD and all FEKs are cleared from memory, closing all applications and leaving all data in non-volatile memory encrypted (a Flush Operation is called once the container is locked).

The TOE intercepts all read/write calls from the Knox Workspace container and automatically encrypts/decrypts the files utilizing the platform encryption libraries.

MOD_FE_V1.0: FDP_PRT_EXT.3

The TOE automatically intercepts all read/write requests for files container within the container boundary. This ensures that any temporary files created by applications within the container boundary are automatically encrypted without the application needing to be made aware.

6.3 Identification and Authentication

PP_APP_V1.3: FIA_AUT_EXT.1
The TOE utilizes the Knox Workspace authentication service to acquire the user password so MKDD can be decrypted and the encrypted data unlocked. The platform provides the password directly to the TOE (without modification). The provided password is conditioned by the TOE with 100,000 rounds of PBKDF2 (using HMAC-SHA-256) to generate a 256-bit KEK.

6.4 Security Management

The TOE is bundled as a component of the Android operating system on supporting Samsung Galaxy devices. The functionality of the TOE is a feature of the Knox Workspace container, and must be enabled via the device management agent when a container is created on the device; the TOE itself does not provide any management capability but relies on the configuration capabilities of the device and Knox Workspace container.

PP_APP_V1.3: FMT_CFG_EXT.1
The TOE relies on the platform (the Knox Workspace container) to handle authentication credentials, and as part of the setup of the Knox Workspace container for the evaluated configuration the user is required to setup their credentials. The TOE relies on Android permissions to restrict access to files created by applications within the Knox Workspace container.

The TOE is a service bundled as part of the Knox Workspace container software on the device. The does not have its own data folder but is a feature within the wider Knox Workspace software. The configuration of the TOE is maintained and protected within the Knox Workspace configuration.

PP_APP_V1.3: FMT_MEC_EXT.1
The TOE configuration is stored as part of the Knox Workspace container configuration and is not maintained separately.

PP_APP_V1.3: FMT_SMF.1
The TOE configuration is limited to the following settings:

- TOE functionality is enabled (it cannot be disabled except by removing the Knox Workspace container)
- Timeout period for automatically entering the Data Lock state after the device has become locked

These settings are specified by the management agent on the device. There are no other configuration settings for the TOE.

MOD_FE_V1.0: FMT_SMF.1(2)
When the Knox Workspace container is removed (such as by unenrollment or by policy by failed login attempts as defined by the Workspace container management), the MKDD will be erased.

6.5 Privacy

PP_APP_V1.3: FPR_ANO_EXT.1

The TOE does not transmit any information over a network.

6.6 Protection of the TSF

PP_APP_V1.3: FPT_AEX_EXT.1
The TOE components are all compiled with the -fstack-protector option set to enable stack-based buffer overflow protection. The Android Linux kernel provides eight unpredictable bits to the base address of any user-space memory mapping (ASLR) which is supported by the parts of the TOE that run User space. For the components that are kernel modules, the kernel supports KASLR to provide unpredictable bits to the base address of the kernel and all its components at startup. The TOE does not invoke READ and WRITE on memory at the same time.

PP_APP_V1.3: FPT_API_EXT.1
The TOE uses only platform provided APIs as noted below:

- All
  - Android SDK API Level 28
  - Samsung Knox API Level 29
- DualDAR Client
  - TEE OS (specific to the device processor) – DRBG
- DualDAR Driver
  - Samsung Android Linux Kernel – DRBG

PP_APP_V1.3: FPT_IDV_EXT.1/PP_APP_V1.3: FPT_TUD_EXT.1
The TOE is part of the Knox Workspace software contained within the Samsung Android operating system. The TOE relies on the platform for all updates as it is considered a part of the operating system. The platform provides the ability to check for and install updates.

The platform user interface provides a method to query the current version of many components, including the TOE software. The TOE software version can be accessed in the Knox version display on the device. This can be found on the *About Phone* section in the Settings.

MOD_FE_V1.0: FPT_KYP_EXT.1
The TOE stores the MKDD and the FEKs separately in non-volatile memory. The MKDD is stored in the TrustZone where it is wrapped with a key derived from the credentials of the Knox Workspace container password (using PBKDF2).

The TOE stores a FEK as part of the file metadata. The FEK is encrypted using AES-GCM with the MKDD KEK.

PP_APP_V1.3: FPT_LIB_EXT.1
The TOE runs on Samsung devices that have launched with at least Android 9 and Knox 3.3. The TOE only utilizes the libraries provided with the platform and does not contain any third party libraries as part of the TOE boundary.

PP_APP_V1.3: ALC_TSU_EXT
Samsung utilizes industry best practices to ensure their devices and all components on the device are patched to mitigate security flaws. Samsung provides a web portal for reporting potential security issues

(https://security.samsungmobile.com/securityReporting.smsb) with instructions about how to securely contact and communicate with Samsung.

Samsung will create updates and patches to resolve reported issues as quickly as possible, at which point the update is provided to the wireless carriers. The delivery time for resolving an issue depends on the severity, and can be as rapid as a few days before the carrier handoff for high priority cases. The wireless carriers perform additional tests to ensure the updates will not adversely impact their networks and then plan device rollouts once that testing is complete. Carrier updates usually take at least two weeks to as much as two months (depending on the type and severity of the update) to be rolled out to customers. However, the Carriers also release monthly Maintenance Releases in order to address security-critical issues, and Samsung itself maintains a security blog (https://security.samsungmobile.com) in order to disseminate information directly to the public.

Samsung communicates with the reporting party to inform them of the status of the reported issue. Further information about updates is handled through the carrier release notes. Issues reported to Google directly are handled through Google's notification processes.

6.7 Trusted Path/Channels

PP_APP_V1.3: FTP_DIT_EXT.1
The TOE does not transmit any data over a network.