KeyW BlackBerry Suite B Data at Rest (ASPP12/ASFEEP10) Security Target

Version 1.0
August 7, 2017

Prepared for:

KeyW Corporation

7880 Milestone Parkway, Suite 100 | Hanover, MD 21076

www.keywcorp.com

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 KeyW BlackBerry Suite B Data at Rest provided by KeyW Corporation. The TOE is being evaluated as a file encryption software 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)

Conventions

The following conventions have been applied in this document:

- Security Functional Requirements (SFR) – Part 2 of the Common Criteria (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 number surrounded by parenthesis and 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, (1) and (2).
    - 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 …").
- The NDPP uses an additional convention – the 'case' – which defines parts of an SFR that apply only when corresponding selections are made or some other identified conditions exist. Only the applicable cases are identified in this ST and they are identified using **bold** text.
- 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 – KeyW BlackBerry Suite B Data at Rest (ASPP12/ASFEEP10) Security Target

ST Version – Version 1.0

ST Date – August 7, 2017

1.2 TOE Reference

TOE Identification – KeyW BlackBerry Suite B Data at Rest, Version 1.2.2.1

TOE Developer – KeyW Corporation

Evaluation Sponsor – United States Special Operations Command (USSOCOM)

1.3 TOE Overview

The Target of Evaluation (TOE) is KeyW BlackBerry Suite B Data at Rest.

The KeyW BlackBerry Suite B Data at Rest TOE is also known as KEYWprotect. The TOE provides an AES-based Data at Rest (DAR) encryption model that is used to encrypt the BlackBerry work space data when BlackBerry 10.3 mobile devices are unlocked including encrypting data received when the BlackBerry work space is locked. The TOE is an application on the BlackBerry 10.3 mobile device. The TOE runs on the following evaluated BlackBerry models.

Device	Processor
Classic	Qualcomm S4 (MSM8960)
Passport	Qualcomm Snapdragon 801
Leap	Qualcomm S4 (MSM8960)
Z30	Qualcomm S4 (MSM8960)
Q10 Porsche	Qualcomm S4 (MSM8960)
Q10	Qualcomm S4 (MSM8960)
Z10 Porsche	Qualcomm S4 (MSM8960)
Z10	Qualcomm S4 (MSM8960)

The TOE utilizes BlackBerry Advanced Data at Rest Protection (ADARP) features to relay all file operations within the BlackBerry work space when the BlackBerry 10.3 mobile device is in the unlocked and locked states. When the BlackBerry 10.3 mobile device is unlocked, all BlackBerry work space data created by other applications is automatically encrypted by the TOE and stored in the BlackBerry work space via the BlackBerry File System (FSYS) Relay API feature. When the BlackBerry 10.3 mobile device is locked, all BlackBerry work space data received by other applications is automatically encrypted by the TOE and stored in the BlackBerry work space via the BlackBerry Data Lock Queue (DLQ) Relay API feature. Encrypted work space data is decrypted as needed only after a user presents valid authentication factors. Therefore, no clear text is ever written to the BlackBerry work space file system.

1.4 TOE Description

The KeyW BlackBerry Suite B Data at Rest application (i.e., the TOE) is a file encryption tool that runs on a BlackBerry 10.3 mobile device. The BlackBerry Advanced Data at Rest Protection (ADARP) relays all operations on files within the BlackBerry work space to the TOE, which encrypts or decrypts the file contents automatically. The TOE runs as a BlackBerry required application, meaning the BlackBerry operating system will ensure that the mobile device features are available only when the TOE is running.

The TOE utilizes the BlackBerry 10.3 operating system for storage of passwords, keys and for Deterministic Random Bit Generation (DRBG). However, the TOE implements its own encryption, decryption, and keyed-Hashing functions, which have been certified through CAVP.

1.4.1 TOE Architecture

KEYWprotect is an application that is installed as a required application when the BlackBerry device is provisioned for use. Being a required application, the BlackBerry 10.3 platform ensures that the TOE is running prior to presenting the home screen to the user, and prevents all actions, which could uninstall the TOE. The TOE provides its own cryptographic functionality, which has been FIPS validated through CAVP certifications (details of the relevant CAVP certificates are provided later in this document).

All attempts to create new files within the BlackBerry work space are relayed to the TOE by Blackberry ADARP and the TOE encrypts these files before they are written to the work space file system. Therefore, no clear text version of the file is ever created on the work space file system.

The TOE generates and stores the following keys and Critical Security Parameters (CSPs) for the File System Relay:

- The TOE generates and stores a 256-bit SALT in the BlackBerry Certificate Manager.
- The TOE derives a 256-bit HASH Key Encryption Key (KEK) from the user's password and 256-bit SALT using PBKDF and stores this KEK in the BlackBerry Certificate Manager.
- The TOE generates and stores a 256-bit Master KEK encrypted by the 256-bit HASH KEK in the BlackBerry Certificate Manager using AES Key Wrap.
- The TOE generates and stores a File Encryption Key (FEK) and a File Authentication Key (FAK) encrypted by the 256-bit Master KEK in the associated metadata of each encrypted file using AES Key Wrap.
- The TOE generates and stores a Message Authentication Code (MAC) immediately following each ciphertext block in each encrypted file.

The TOE generates and stores the following keys and Critical Security Parameters (CSPs) for the Data Lock Queue Relay:

- The TOE generates and stores a static 521-bit ECC DLQ Master encryption keypair in the BlackBerry Certificate Manager.
- The TOE generates and stores the following keys and CSPs for each open/write of a file or after crossing the 1 megabyte Anchor Point boundary, which represents a starting point position in the file for encrypting data and therefore also the position where data will be decrypted:
    - The TOE generates an ephemeral 521-bit ECC Anchor Point keypair.
    - The TOE computes a shared secret using ECC CDH Primitive from the 521-bit ECC DLQ Master encryption keypair and ephemeral 521-bit ECC Anchor Point keypair.
    - The TOE derives a 256-bit Root Encryption Key (REK) from the shared secret using SHA.
    - The TOE derives an ephemeral 521-bit ECC DLQ keypair, static 521-bit ECC Anchor Point keypair, 36-byte Nonce, and a 128-bit IV from the REK using a KBKDF.
        - The TOE stores the 128-bit IV in the Anchor Point context contained in volatile memory.
        - The TOE stores the compressed ephemeral 521-bit ECC Anchor Point in the associated metadata of the Anchor Point.
    - The TOE computes a 256-bit MAC key, a 256-bit Key Encryption Key (KEK), a 256-bit Data Encryption Key (DEK), and a 256-bit MAC tag using ECC KAS.
        - The TOE stores the 256-bit KEK in the Anchor Point context contained in volatile memory when processed as the Master Anchor Point context.
        - The TOE stores the 256-bit DEK in the Anchor Point context contained in volatile memory where the DEK is encrypted by a 256-bit Master Anchor Point context KEK for only decrypt operations.
        - The TOE stores the 256-bit MAC key encrypted by the REK using AES Key Wrap, and the 256-bit MAC tag in the associated metadata of the Anchor Point.
- The TOE chains forward the 256-bit DEK and 128-bit IV for each 4 kilobytes of file data using a KBKDF where the DEK is encrypted by a 256-bit Master Anchor Point context KEK for only decrypt operations.
- The TOE stores the chain counter in the Anchor Point context contained in volatile memory.

1.4.1.1 Physical Boundaries

The physical boundary of the TOE is the physical perimeter of the evaluated device (BlackBerry 10.3) on which the TOE resides.

1.4.1.2 Logical Boundaries

This section summarizes the security functions provided by KEYWprotect.

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

1.4.1.2.1 Cryptographic support

The TOE operates on a BlackBerry 10.3 mobile device and uses features provided by the platform for key storage, user credential storage, and Deterministic Random Bit Generation (DRBG). The TOE implements its own algorithms for AES, Key Wrapping, key-based and password-based Key Derivation, Key Establishment, cryptographic hashing and keyed-hashing.

1.4.1.2.2 User data protection

The TOE protects user data by providing an integrated file encryption and file data authentication capability that automatically encrypts new files and decrypts files upon user demand. The TOE utilizes 256-bit AES encryption for confidentiality and HMAC-SHA-384 for file data integrity.

1.4.1.2.3 Identification and authentication

The TOE authenticates a user by requiring a password before any file data decryption operation is initiated. Without the correct password, the user is unable to decrypt the keys necessary to obtain clear text data from the BlackBerry work space file system.

1.4.1.2.4 Security management

The TOE supports encryption while in the locked state, but does not allow decryption or integrity operations until the user authenticates to the device upon first use of the TOE. The TOE allows the user to change their password for management purposes.

1.4.1.2.5 Protection of the TSF

The TOE relies on the physical boundary of the evaluated platform as well as the BlackBerry 10.3 operating system for the protection of the TOE's application components.

Updates to the TOE are handled by the BlackBerry Enterprise Services (BES) management software.

1.4.1.2.6 Trusted path/channels

The TOE does not transmit any data between itself and another network entity. All of the data managed by the TOE resides on the evaluated platform (BlackBerry 10.3).

1.4.2 TOE Documentation

KeyW offers the following documentation to users for the installation and operation of their product. The following list of documents was examined as part of the evaluation.

[Guide] KeyW BB10 Suite B Data At Rest v1.2.2.1 – User Guide, Document Version 1.1, July 21, 2017.

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 4, September 2012.
    - Part 2 Extended
- Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, Version 3.1 Revision 4, September 2012.
    - Part 3 Extended
- Package Claims:
    - Protection Profile for Application Software, Version 1.2, 22 April 2016 and Application Software Protection Profile (ASPP12)
    - Extended Package: File Encryption: Mitigating the Risk of Disclosure of Sensitive Data on a System, Version 1.0, 10 November 2014 (ASFEEP10)

2.1 Conformance Rationale

The ST conforms to the ASPP12/ASFEEP10. As explained previously, the security problem definition, security objectives, and security requirements have been drawn from the PP.

3. Security Objectives

The Security Problem Definition may be found in the ASPP12/ASFEEP10 and this section reproduces only the corresponding Security Objectives for operational environment for reader convenience. The ASPP12/ASFEEP10 offers additional information about the identified security objectives, but that has not been reproduced here and the ASPP12/ASFEEP10 should be consulted if there is interest in that material.

In general, the ASPP12/ASFEEP10 has defined Security Objectives appropriate for file encryption software application and as such are applicable to the KeyW BlackBerry Suite B Data at Rest TOE.

3.1 Security Objectives for the Operational Environment

OE.AUTHORIZATION_FACTOR_STRENGTH 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 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 The non-mobile operational environment must be configurable so that there exists at least one mechanism that will cause the system to power down after a period of time in the same fashion as the user electing to shutdown the system (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_ADMIN 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.PROPER_USER 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.STRONG_ENVIRONMENT_CRYPTO The Operating environment will provide a cryptographic function capability that is commensurate with the requirements and capabilities of the TOE.

OE.TRAINED_USERS Authorized users of the host machine will be trained to follow all provided guidance.

4. Extended Components Definition

All of the extended requirements in this ST have been drawn from the ASPP12/ASFEEP10. The ASPP12/ASFEEP10 defines the following extended requirements and since they are not redefined in this ST the ASPP12/ASFEEP10 should be consulted for more information in regard to those CC extensions.

Extended SFRs:

- FCS_CKM_EXT.1: Key Encrypting Key (KEK) Support
- FCS_CKM_EXT.2: Cryptographic key generation (FEK)
- FCS_CKM_EXT.4: Extended: Cryptographic Key Destruction
- FCS_CKM_EXT.5: File Authentication Key (FAK) Support
- FCS_IV_EXT.1: Extended: Initialization Vector Generation
- FCS_KYC_EXT.1: Key Chaining and Key Storage
- FCS_RBG_EXT.1: Random Bit Generation Services
- FCS_STO_EXT.1: Storage of Secrets
- FDP_AUT_EXT.1: Extended: Authentication of Selected User Data
- FDP_AUT_EXT.2: Extended: Data Authentication using cryptographic, keyed hash functions
- FDP_DAR_EXT.1: Encryption Of Sensitive Application Data
- FDP_DEC_EXT.1: Access to Platform Resources
- FDP_PM_EXT.1: Extended: Protection of Data in Power Managed States
- FDP_PRT_EXT.1: Extended: Protection of Selected User Data
- FIA_AUT_EXT.1: User Authorization
- FIA_FCT_EXT.1(2): Extended: User Authorization with Password/Passphrase Authorization Factors
- FMT_CFG_EXT.1: Secure by Default Configuration
- FMT_MEC_EXT.1: Supported Configuration Mechanism
- FPT_AEX_EXT.1: Anti-Exploitation Capabilities
- FPT_API_EXT.1: Use of Supported Services and APIs
- FPT_FEK_EXT.1: File Encryption Key (FEK) Support
- FPT_KYP_EXT.1: Extended: Protection of Key and Key Material (FPT_KYP_EXT)
- FPT_LIB_EXT.1: Use of Third Party Libraries
- FPT_TUD_EXT.1: Integrity for Installation and Update
- FTP_DIT_EXT.1: Protection of Data in Transit

Extended SARs:

- ALC_TSU_EXT.1: Timely Security Updates

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 ASPP12/ASFEEP10. The refinements and operations already performed in the ASPP12/ASFEEP10 are not identified (e.g., highlighted) here, rather the requirements have been copied from the ASPP12/ASFEEP10 and any residual operations have been completed herein. Of particular note, the ASPP12/ASFEEP10 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 ASPP12/ASFEEP10 which includes all the SARs for EAL 1. However, the SARs are effectively refined since requirement-specific 'Assurance Activities' are defined in the ASPP12/ASFEEP10 that serve to ensure corresponding evaluations will yield more practical and consistent assurance than the EAL 1 assurance requirements alone. The ASPP12/ASFEEP10 should be consulted for the assurance activity definitions.

5.1 TOE Security Functional Requirements

The following table identifies the SFRs that are satisfied by KeyW BlackBerry Suite B Data at Rest TOE.

Requirement Class	Requirement Component
FCS: Cryptographic support	FCS_CKM.1(A): Cryptographic key generation (Password/Passphrase conditioning)
	FCS_CKM_EXT.1: Key Encrypting Key (KEK) Support
	FCS_CKM_EXT.1(A): Extended: Cryptographic key generation (Password/Passphrase conditioning) (TD0067 applied)
	FCS_CKM_EXT.2: Cryptographic key generation (FEK)
	FCS_CKM_EXT.4: Extended: Cryptographic Key Destruction
	FCS_COP.1(1): Cryptographic operation (Data Encryption)
	FCS_COP.1(4): Cryptographic Operation (Keyed-Hash Message Authentication)
	FCS_COP.1(5): Cryptographic operation (Key Wrapping)
	FCS_COP.1(6): FAK encryption/decryption support
	FCS_IV_EXT.1: Extended: Initialization Vector Generation
	FCS_KYC_EXT.1: Key Chaining and Key Storage
	FCS_RBG_EXT.1: Random Bit Generation Services
	FCS_STO_EXT.1: Storage of Credentials
FDP: User data protection	FDP_AUT_EXT.1: Extended: Authentication of Selected User Data
	FDP_AUT_EXT.2: Extended: Data Authentication using cryptographic, keyed hash functions
	FDP_DAR_EXT.1: Encryption Of Sensitive Application Data
	FDP_DEC_EXT.1: Access to Platform Resources
	FDP_NET_EXT.1: Network Communications
	FDP_PM_EXT.1: Extended: Protection of Data in Power Managed States
	FDP_PRT_EXT.1: Extended: Protection of Selected User Data
FIA: Identification and authentication	FIA_AUT_EXT.1: User Authorization
	FIA_FCT_EXT.1(2): Extended: User Authorization with Password/Passphrase Authorization Factors
FMT: Security management	FMT_CFG_EXT.1: Secure by Default Configuration
	FMT_MEC_EXT.1: Supported Configuration Mechanism
	FMT_SMF.1: Specification of Management Functions

FPR: Privacy	FPR_ANO_EXT.1: User Consent for Transmission of Personally Identifiable
FPT: Protection of the TSF	FPT_AEX_EXT.1: Anti-Exploitation Capabilities
	FPT_API_EXT.1: Use of Supported Services and APIs
	FPT_FEK_EXT.1: File Encryption Key (FEK) Support
	FPT_KYP_EXT.1: Extended: Protection of Key and Key Material (FPT_KYP_EXT)
	FPT_LIB_EXT.1: Use of Third Party Libraries
	FPT_TUD_EXT.1: Integrity for Installation and Update
FTP: Trusted path/channels	FTP_DIT_EXT.1: Protection of Data in Transit

Table 1 TOE Security Functional Components

5.1.1 Cryptographic support (FCS)

5.1.1.1 Cryptographic key generation (Password/Passphrase conditioning) (FCS_CKM.1(A))

FCS_CKM.1(A).1

Requirement renamed to FCS_CKM_EXT.1(A) per TD0067.

5.1.1.2 Extended: Cryptographic key generation (Password/Passphrase conditioning) (TD0067 applied) (FCS_CKM_EXT.1(A))

FCS_CKM_EXT.1(A).1

The TSF shall support a password/passphrase of up to [**64**] characters used to generate a password authorization factor.

FCS_CKM_EXT.1(A).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 [***no other characters***].

FCS_CKM_EXT.1(A).3

The TSF shall perform Password-based Key Derivation Functions in accordance with a specified cryptographic algorithm HMAC-[***SHA-384***], with [**12345**] iterations, and output cryptographic key sizes [***256***] that meet the following: NIST SP 800-132.

FCS_CKM_EXT.1(A).4

The TSF shall not accept passwords less than [***14***] and greater than the maximum password length defined in FCS_CKM_EXT.1(A).1.

FCS_CKM_EXT.1(A).5

The TSF shall generate all salts using a RBG that meets FCS_RBG_EXT.1 (from the AS PP) and with entropy corresponding to the security strength selected for PBKDF in FCS_CKM_EXT.1(A).3. (Renumbered by TD0067)

5.1.1.3 Key Encrypting Key (KEK) Support (FCS_CKM_EXT.1)

FCS_CKM_EXT.1.1

The TSF shall support KEK in the following manner based on the selection chosen in FPT_FEK_EXT.1: [***derive a KEK using a password-based authorization factor conditioned as defined in FCS_CKM_EXT.1(A) and in accordance with FIA_FCT_EXT.1(2), using a Random Bit Generator as specified in FCS_RBG_EXT.1 (from the AS PP) and with entropy corresponding to the security strength of AES key sizes of [256 bit]***].

FCS_CKM_EXT.1.2

All KEKs shall be [***256-bit***] keys corresponding to at least the security strength of the keys encrypted by the KEK.

5.1.1.4 Cryptographic key generation (FEK) (FCS_CKM_EXT.2)

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 the AS PP) and with entropy corresponding to the security strength of AES key sizes of [256 bit]***]

FCS_CKM_EXT.2.2

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

FCS_CKM_EXT.2.3

The FEKs must be generated by the TOE.

5.1.1.5 Extended: Cryptographic Key Destruction (FCS_CKM_EXT.4)

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]]]

that meets the following: No Standard. (TD0175 applied).

5.1.1.6 File Authentication Key (FAK) Support (FCS_CKM_EXT.5)

FCS_CKM_EXT.5.1

The TSF shall use a FAK to authenticate sensitive data when a cryptographic, keyed hashing function is used for data authentication and shall be supported in the following manner: [***a FAK will be stored in non-volatile memory encrypted with a KEK as specified in FCS_COP.1(5) using authorization factors as specified in FCS_CKM_EXT.1***].

FCS_CKM_EXT.5.2

The TSF shall create a unique FAK for each file (or set of files) using the mechanism on the client as specified in FCS_RBG_EXT.1.

FCS_CKM_EXT.5.3

The FAKs must be generated by the TOE.

FCS_CKM_EXT.5.4

The TSF will not write FAKs to non-volatile memory.

FCS_CKM_EXT.5.5

The FAK shall be protected in a manner conformant to FCS_COP.1(6).

5.1.1.7 Cryptographic operation (Data Encryption) (FCS_COP.1(1))

FCS_COP.1(1).1

Refinement: The application shall [***implement AES encryption***] to perform data encryption and decryption in accordance with a specified cryptographic algorithm AES used in [***CBC (as defined in NIST SP 800-38A), XTS (as defined in NIST SP 800-38E)***] mode and cryptographic key sizes [***256 bits***].

5.1.1.8 Cryptographic Operation (Keyed-Hash Message Authentication) (FCS_COP.1(4))

FCS_COP.1(4).1

Refinement: The application shall [***implement functionality***] to perform keyed-hash message authentication in accordance with a specified cryptographic algorithm HMAC- [***SHA-384***] , key size [***256***] , and message digest size of [***384***] bits that meet the following: FIPS PUB 198-1, 'The Keyed-Hash Message Authentication Code', and FIPS PUB 180-4, 'Secure Hash Standard'.

5.1.1.9 Cryptographic operation (Key Wrapping) (FCS_COP.1(5))

FCS_COP.1(5).1

Refinement: The application shall [***implement functionality to perform Key Wrapping***] in accordance with a specified cryptographic algorithm [***AES Key Wrap***] and the cryptographic key size [***256 bits (AES)***] that meet the following: [***NIST SP 800-38F for Key Wrap (section 6.2) and Key Wrap with Padding (section 6.3)***].

5.1.1.10 FAK encryption/decryption support (FCS_COP.1(6))

FCS_COP.1(6).1

The FAK shall be protected in the same manner as the FEK, in accordance with FCS_COP.1(5).

5.1.1.11 Extended: Initialization Vector Generation (FCS_IV_EXT.1)

FCS_IV_EXT.1.1

The application shall [***generate IVs***] in accordance with Appendix H: Initialization Vector Requirements for NIST-Approved Cipher Modes.

5.1.1.12 Key Chaining and Key Storage (FCS_KYC_EXT.1)

FCS_KYC_EXT.1.1

The TSF shall maintain a primary key chain of [

- ***KEKs originating from one or more authorization factor(s) to the FEK(s) using the following method(s): [utilization of the platform key storage], while maintaining an effective strength of [[256 bits] for symmetric keys] commensurate with the strength of the FEK***].

and [***No supplemental key chains***]. (TD0076, TD0092 and TD0123 applied)

5.1.1.13 Random Bit Generation Services (FCS_RBG_EXT.1)

FCS_RBG_EXT.1.1

The application shall [***invoke platform-provided DRBG functionality***] for its cryptographic operations.

5.1.1.14 Storage of Credentials (FCS_STO_EXT.1)

FCS_STO_EXT.1.1

The application shall [***invoke the functionality provided by the platform to securely store [user password]***] to non-volatile memory.

5.1.2 User data protection (FDP)

5.1.2.1 Extended: Authentication of Selected User Data (FDP_AUT_EXT.1)

FDP_AUT_EXT.1.1

The TSF shall perform authentication of the user-selected file (or set of files) and provide notification to the user if modification had been detected.

FDP_AUT_EXT.1.2

The TSF shall implement a data authentication method based on [***cryptographic, keyed hashing service and verification in accordance with FDP_AUT_EXT.2***].

5.1.2.2 Extended: Data Authentication using cryptographic, keyed hash functions (FDP_AUT_EXT.2)

FDP_AUT_EXT.2.1

The TSF shall use a cryptographic, keyed hash function in accordance with FCS_COP.1(4).

FDP_AUT_EXT.2.2
The TSF shall use a File Authentication Key (FAK) in accordance with FCS_COP.1(6) and FCS_CKM_EXT.5 as the secret key to the keyed hash function.

FDP_AUT_EXT.2.3
The TSF shall use the entirety of the ciphertext file as the message input to the keyed hash function.

FDP_AUT_EXT.2.4
The TSF shall concatenate the output of the keyed hash function, the Message Authentication Code (MAC).

FDP_AUT_EXT.2.5
The TSF shall authenticate the encrypted file prior to decryption.

FDP_AUT_EXT.2.6
The TSF shall authenticate the data by comparing the keyed hash output of the ciphertext against the stored MAC.

FDP_AUT_EXT.2.7
The TSF shall notify the user of an unsuccessful authentication and prevent decryption of the ciphertext.

FDP_AUT_EXT.2.8
During verification, the TSF shall assume the MAC is at the end of the ciphertext file.

FDP_AUT_EXT.2.9
The FAK will be generated using a RBG that meets FCS_RBG_EXT.1 (from the AS PP).

5.1.2.3 Encryption Of Sensitive Application Data (FDP_DAR_EXT.1)

FDP_DAR_EXT.1.1
The application shall [***implement functionality to encrypt sensitive data***] in non-volatile memory.

5.1.2.4 Access to Platform Resources (FDP_DEC_EXT.1)

FDP_DEC_EXT.1.1:
The application shall restrict its access to [***/LED/***].

FDP_DEC_EXT.1.2:
The application shall restrict its access to **[*/shared files/*].**

5.1.2.5 Network Communications (FDP_NET_EXT.1)

FDP_NET_EXT.1.1
The application shall restrict network communication to [***no network communication***].

5.1.2.6 Extended: Protection of Data in Power Managed States (FDP_PM_EXT.1)

FDP_PM_EXT.1.1
The TSF shall protect all data stored to the disk drive 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.

5.1.2.7 Extended: Protection of Selected User Data (FDP_PRT_EXT.1)

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).

FDP_PRT_EXT.1.2
The application 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. (TD0065 applied)

5.1.3 Identification and authentication (FIA)

5.1.3.1 User Authorization (FIA_AUT_EXT.1)

FIA_AUT_EXT.1.1

The application shall [***provide user authorization***] based on [***password/passphrase authorization factors***].

5.1.3.2 Extended: User Authorization with Password/Passphrase Authorization Factors (FIA_FCT_EXT.1(2))

FIA_FCT_EXT.1(2).1

The TSF shall provide a mechanism as defined in FCS_CKM_EXT.1 and FCS_COP.1(4) to perform user authorization.

FIA_FCT_EXT.1(2).2

The TSF shall perform user authorization using the mechanism provided in FIA_FCT_EXT.1.1(2) before allowing decryption of user data.

FIA_FCT_EXT.1(2).3

The TSF shall support the use of multiple instances of authorization factors that result in unique encryption keys.

FIA_FCT_EXT.1(2).4

The TSF shall verify that the user-entered authorization factors are valid before decrypting the user's encrypted files.

FIA_FCT_EXT.1(2).5

The TSF shall ensure that the method of validation for each authorization factor does not expose or reduce the effective strength of the KEK, FEK, or CSPs used to derive the KEK or FEK.

FIA_FCT_EXT.1(2).6

The TSF shall perform user authorization using the mechanism provided in FIA_FCT_EXT.1.1(2) before allowing the user to change the passphrase-based authorization factor as specified in FMT_SMF.1(c).

5.1.4 Security management (FMT)

5.1.4.1 Secure by Default Configuration (FMT_CFG_EXT.1)

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 it and its data from unauthorized access.

5.1.4.2 Supported Configuration Mechanism (FMT_MEC_EXT.1)

FMT_MEC_EXT.1.1

The application shall invoke the mechanisms recommended by the platform vendor for storing and setting configuration options.

5.1.4.3 Specification of Management Functions (FMT_SMF.1)

FMT_SMF.1.1

The TSF shall be capable of performing the following management functions: [***change password/passphrase authentication factors***]. (TD0221 applied)

5.1.5 Privacy (FPR)

5.1.5.1 User Consent for Transmission of Personally Identifiable Information (FPR_ANO_EXT.1)

FPR_ANO_EXT.1.1
The TSF shall [***not transmit PII over a network***].

5.1.6 Protection of the TSF (FPT)

5.1.6.1 Anti-Exploitation Capabilities (FPT_AEX_EXT.1)

FPT_AEX_EXT.1.1
The application shall not request to map memory at an explicit address except for [**none**].
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 compiled with stack-based buffer overflow protection enabled.

5.1.6.2 Use of Supported Services and APIs (FPT_API_EXT.1)

FPT_API_EXT.1.1
The application shall only use documented platform APIs.

5.1.6.3 File Encryption Key (FEK) Support (FPT_FEK_EXT.1)

FPT_FEK_EXT.1.1
The TSF shall [***Store a FEK in Non-Volatile memory conformant with FPT_KYP_EXT.1***].

5.1.6.4 Extended: Protection of Key and Key Material (FPT_KYP_EXT) (FPT_KYP_EXT.1)

FPT_KYP_EXT.1.1
The TSF shall [***only store keys in non-volatile memory when [wrapped as specified in FCS_COP.1(5)]***]. (TD0123 applied)

5.1.6.5 Use of Third Party Libraries (FPT_LIB_EXT.1)

FPT_LIB_EXT.1.1
The application shall be packaged with only [**BlackBerry Platform libraries**].

5.1.6.6 Integrity for Installation and Update (FPT_TUD_EXT.1)

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 be distributed using the format of the platform-supported package manager.
FPT_TUD_EXT.1.3
The application shall be packaged such that its removal results in the deletion of all traces of the application, with the exception of configuration settings, output files, and audit/log events.
FPT_TUD_EXT.1.4
The application shall not download, modify, replace or update its own binary code.

FPT_TUD_EXT.1.5

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

FPT_TUD_EXT.1.6

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

5.1.7 Trusted path/channels (FTP)

5.1.7.1 Protection of Data in Transit (FTP_DIT_EXT.1)

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 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
	ALC_TSU_EXT.1: Timely Security Updates
ATE: Tests	ATE_IND.1: Independent testing - conformance
AVA: Vulnerability assessment	AVA_VAN.1: Vulnerability survey

Table 2 **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 fulfil 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 labeled 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.

5.2.3.3 Timely Security Updates (ALC_TSU_EXT.1)

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.1.2d
The developer shall provide a description in the TSS of how users are notified when updates change security properties or the configuration of the product.

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.1.1e
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 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.

5.2.5 Vulnerability assessment (AVA)

5.2.5.1 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
- Identification and authentication
- Security management
- Protection of the TSF
- Trusted path/channels

6.1 Cryptographic support

The TOE operates on a BlackBerry 10.3 mobile device and uses features provided by the platform for keys storage, user credential storage, and Deterministic Random Bit Generation (DRBG). The BlackBerry mobile device utilizes the Qualcomm Snapdragon S4 and Qualcomm Snapdragon 801 processors having the ARMv7 architecture.

The TOE implements its own algorithms for AES, Key Wrapping, key-based and password-based Key Derivation, Key Establishment, cryptographic hashing and keyed-hashing by supporting the following algorithms:

Functions	Standards	Certificates
Encryption/Decryption		
• AES CBC (256 bits)	FIPS PUB 197 NIST SP 800-38A	AES Certificate # 4312
Encryption/Decryption		
• AES XTS (256 bits)	FIPS PUB 197 NIST SP 800-38E	AES Certificate # 3328
Cryptographic hashing		
• SHA-384	FIPS Pub 180-4	SHS Certificate # 2761
Keyed-hash message authentication		
• HMAC-SHA-384	FIPS Pub 198-1 FIPS Pub 180-4	HMAC Certificate # 2119
Key Wrapping		
• AES Key Wrap (256-bit)	NIST SP 800-38F	AES Certificate # 3328
Password-Based Key Derivation		
• HMAC-SHA-384	NIST SP 800-132	Vendor-Affirmed

The Cryptographic support function is designed to satisfy the following security functional requirements:

- FCS_CKM_EXT.1(A): A user password prompt must be presented to the user immediately following the installation of the TOE on the BlackBerry platform and prior to decryption of any file. The password is used to derive a 256-bit HASH KEK. This KEK is derived by applying a PBKDFv2 mechanism to the password, meeting SP 800-132. This derivation uses an HMAC-SHA-384 algorithm, 12345 iterations, and a 256-bit SALT value to create the 256-bit HASH KEK. The SALT value is obtained from the BlackBerry DRBG. The TOE support passwords having up to 64 characters maximum in length, but also supports passwords as low as 14 characters in length. The 14 character minimum password length and 64 character maximum length is enforced by the TOE.
- FCS_CKM_EXT.1: The TOE utilizes the user's password to derive a 256-bit HASH KEK as described above and obtains a 256-bit Master KEK from the BlackBerry DRBG. The 256-bit HASH KEK and 256-bit Master KEK are stored in the BlackBerry Certificate Manager. The 256-bit Master KEK is stored encrypted by the 256-bit HASH KEK.

- FCS_CKM_EXT.2: The TOE automatically generates a new FAK and FEK whenever the application creates a new file. The FAK and FEK never change during the life of a file. The TOE obtains 256-bits of random bits from the BlackBerry DRBG, which become the new FEK and FAK for the file.
- FCS_CKM_EXT.4: User passwords temporarily stored in volatile memory are cleared by invoking platform functionality. Passwords that have been stored in the BlackBerry Credential Manager are cleared and erased by invoking the appropriate Credential Manager API. When a password is obtained by the TOE application from the BlackBerry Credential Manager, the TOE uses the Credential Manager API to clear the buffer holding the password. The password is no longer needed in volatile memory once the 256-bit HASH KEK can be derived from the password.

  When the TOE application obtains a KEK from the BlackBerry Certificate Manager, the TOE implements a read/verify after zeroing volatile memory into which the KEK was read. The TOE invokes the appropriate BlackBerry Certificate Manager API to clear a KEK when it is no longer needed. The 256-bit HASH KEK is no longer needed in volatile memory once the application is "Data Locked". The 256-bit Master KEK is no longer needed in volatile memory once the application has completed using it to unwrap the FEK and FAK of a file. The 256-bit HASH KEK is no longer needed in the BlackBerry Certificate Manager upon rekey, which occurs once the user changes the password.
- FCS_CKM_EXT.5: The TOE associates a FAK generated from the BlackBerry DRBG with each encrypted file. The FAK uses HMAC-SHA-384 to calculate a cryptographic hash of the ciphertext of a file. This FAK is wrapped by the 256-bit Master KEK and stored in the associated metadata of each encrypted file, while the clear text, calculated hash value is stored following each ciphertext block in each encrypted file. The TOE never writes the clear text FAK to a non-volatile storage medium.
- FCS_COP.1(1): The TOE implements its own AES in CBC and XTS modes, with key sizes of 256 bits as defined by NIST SP 800-38A (refer to CAVS Certificate # 4312) and NIST SP 800-38E (refer to CAVS Certificate # 3328). Users do not choose an encryption mode, within the File System Relay all FEKs in the system are 256-bit AES-XTS mode keys and within the Data Lock Queue Relay all DEKs in the system are 256-bit AES-CBC mode keys.
- FCS_COP.1(4): The TOE implements its own keyed-Hash for HMAC-SHA-384 that is compliant with FIPS Pub 180-4 and FIPS Pub 198-1 (CAVS Certificate # 2119). The TOE performs HMAC-SHA-384 on 32-kilobyte blocks, using SHA-384, with a key length of 256 and output length of 384.
- FCS_COP.1(5): The TOE performs key wrapping of the 256-bit Master KEK, FEK and FAK using an AES 256 bit key wrap conformant to NIST SP 800-38F (refer to CAVS Certificate # 3328).
- FCS_COP.1(6): The TOE performs key wrapping of the FAK using an AES Key wrap (refer to CAVS Certificate #3328).
- FCS_IV_EXT.1: The TOE generates Initialization Vectors (IV) using the AES CTR-DRBG provided by the BlackBerry DRBG. The TOE uses IVs when encrypting file data using a FEK.
- FCS_KYC_EXT.1: The 256-bit Master KEK, and FEK are both 256-bit AES keys, protected using an AES Key Wrap when stored. The 256-bit Master KEK is stored in the BlackBerry Certificate Manager wrapped by the 256-bit HASH KEK derived from the user's password (using PBKDF2). The FEK is stored in the associated metadata of the file it protects (and before the wrapped FAK). The FEK is stored only after being AES Key Wrapped by the 256-bit Master KEK.
- FCS_RBG_EXT.1: The TOE uses the BlackBerry DRBG source that is available through the Security Builder API. The Security Builder API is configured for AES CTR-DRBG.
- FCS_STO_EXT.1: The TOE stores the user's password in the BlackBerry Credential Manager to authenticate the user, thereby providing access to the File Authentication Key (FAK) and File Encryption Key (FEK).

6.2 User data protection

The User data protection function is designed to satisfy the following security functional requirements:

- FDP_AUT_EXT.1: The TOE associates a FAK generated from the BlackBerry DRBG with each encrypted file. The FAK uses HMAC-SHA-384 to calculate a cryptographic hash of the ciphertext of a file. Before the TOE decrypts a file, the HMAC operation is performed on the ciphertext blocks of a file. Thus, once the file is authenticated by the FAK it will then be decrypted by the FEK using an AES cryptographic operation. If the FAK results in a cryptographic hash that does not match the saved HASH, the user is notified of the failure and prevents decryption of the ciphertext block.
- FDP_AUT_EXT.2: The TOE uses HMAC-SHA-384 as a cryptographic keyed-hash to detect modifications to encrypted file data. The FAK is used to calculate a hash to verify the integrity of the encrypted file data. The FAK and the FEK are used authenticate and decrypt the file. The FAK is used to calculate a keyed-hash over the entire ciphertext of the file and results in a message authentication code for each ciphertext block which is appended to the end of each ciphertext block. If the calculated MAC does not match the saved MAC when a ciphertext block of an encrypted file is being authenticated then the ciphertext block is not decrypted and the user is notified of the failure. The FAK is obtained by reading 256 bits from the BlackBerry DRBG.
- FDP_DAR_EXT.1: The TOE provides its own CAVP certified cryptographic algorithms to encrypt all files in the mobile device's work space directory structure.
- FDP_DEC_EXT.1: The typical installation method for this application is via a Mobile Device Management (MDM) system. The BlackBerry MDM is known as BlackBerry Enterprise Services (BES). A device is provisioned by BES and the TOE application is installed with a 'required' disposition. This means the application is installed automatically and will always be executing at startup before the mobile device presents the home screen. Upon installation, the TOE indicates its intent to access LED, shared files, run in the background, and access device identifying information. The TOE does not transmit any information over a network.
- FDP_NET_EXT.1: The TOE does not transmit any information over a network.
- FDP_PM_EXT.1: The "Data Locked" state is when the screen is locked, password locked or inactivity period lapsed. In this state the TOE deletes the 256-bit HASH KEK and encrypted 256-bit Master KEK from volatile memory, leaving only the encrypted files. A screen can be screen locked, password locked, or data locked after a period of inactivity configured in the BlackBerry platform (using the BES management system). The TOE requires the user to re-authenticate (i.e., provide the password) to decrypt files following a BlackBerry entering the "Data Locked" state.

  When a device enters the “Data Locked” state, the TOE deletes the 256-bit HASH KEK and encrypted 256-bit Master KEK from volatile memory. If a power failure occurs, the 256-bit HASH KEK and encrypted 256-bit Master KEK are cleared from volatile memory automatically. Because the TOE is installed as a mandatory service, the user's home screen is not displayed until the TOE has completed its initialization.
- FDP_PRT_EXT.1: The TOE automatically encrypts all files stored in the BlackBerry work space directories upon the file's creation. No temporary files are used during the encryption/decryption process as all file content processing is done with in-place buffers. A password is required in order to decrypt the contents of a file. The TOE encrypts only the file contents and not the file metadata. The BlackBerry Advanced Data at Rest Protection (ADARP) relays all operations on files within the BlackBerry work space to the TOE, which encrypts or decrypts the file contents automatically. When the device data locked, as described above, the TOE encrypts all work space data received by applications, so no clear text version of the files are ever created on the work space file system. When the device is data unlocked, the TOE encrypts all work space files as they are created by other applications.

6.3 Identification and authentication

The Identification and authentication function is designed to satisfy the following security functional requirements:

- FIA_AUT_EXT.1: The TOE utilizes BlackBerry provided APIs to request credentials from users, and to store/retrieve those credentials in the BlackBerry Credential Manager. The TOE performs a comparison and

makes its own decisions regarding the correctness of the password provided by the user. The TOE treats the password as a case-sensitive comparison.

- FIA_FCT_EXT.1(2): Prior to allowing any decryption of file data, the TOE obtains a password from the mobile device user. The TOE then uses the password and a SALT value (saved in the BlackBerry Certificate Manager), with 12345 iterations of a PBKDF2 algorithm, to derive a 256-bit HASH KEK. This 256-bit HASH KEK is then used to unwrap the 256-bit Master KEK that was stored in the BlackBerry Certificate Manager. If the presented password is invalid, the resulting 256-bit HASH KEK will be unable to unwrap the Master KEK. Thus, a change to a user's password requires the old password to unwrap the Master KEK, and a new password to provide the new wrap for the Master KEK.

6.4 Security management

The Security management function is designed to satisfy the following security functional requirements:

- FMT_CFG_EXT.1: After the TOE application has been installed, the mobile device user is prompted to define the first user password. The TOE supports only a single mobile device user having one password. The user may change the password through the TOE interface. In BlackBerry, a mobile device user has RW access to the directories containing shared files, thus allowing the application to have RW permission to the directories containing shared files.
- FMT_MEC_EXT.1: The TOE uses a very limited set of configuration data. This includes only the user's password, along with a value representing the count of failed authentication attempts.
- FMT_SMF.1: The TOE allows a user to change the password used to authenticate the user and protect the 256-bit Master KEK. The TOE requires both the old and new password during this change, unwraps the 256-bit Master KEK using the old password and wraps the 256-bit Master KEK using the new password.

6.5 Privacy

The Privacy function is designed to satisfy the following security functional requirements:

- FPR_ANO_EXT.1: The TOE does not transmit any information over a network.

6.6 Protection of the TSF

The Protection of the TSF function is designed to satisfy the following security functional requirements:

- ALC_TSU_EXT.1: KeyW accepts bug reports (including reports for security vulnerabilities) through a Technical Support Contact form on the www.keywcorp.com/contact-us/ web site. KeyW reviews all bug reports when making product changes to resolve issues associated with the TOE. KeyW makes updates and code patches to resolve issues as quickly as possible, and makes updates available to customers. TOE updates are distributed by KeyW to customers, which can then utilize the BES management system to distribute the update to individual devices.
- FPT_AEX_EXT.1: The TOE does not map memory to a specific address, does not allocate memory with both write and execute permission and does not write user-modifiable files to locations which contain executables. The application is compiled with FSTACK protection STRONG. The TOE executes on mobile devices running BlackBerry 10.3.3 and utilizes the security features from this BlackBerry platform.
- FPT_API_EXT.1: The TOE utilizes the APIs identified throughout the TSS and those listed in Section 7.
- FPT_FEK_EXT.1: An AES Key Wrapped FEK associated with a file is stored in the associated metadata of the file.
- FPT_KYP_EXT.1: All FEK are wrapped using AES Key Wrap with a 256-bit key, prior to being stored with the file the FEK is protecting. The FEK is never stored in non-volatile memory in clear text form.
- FPT_LIB_EXT.1: The TOE runs on the BlackBerry 10.3.3 platform, utilizing libraries provided by the platform. The TOE does not utilize any other 3rd party libraries.

- FPT_TUD_EXT.1: The TOE application software is distributed and installed as an "internal app" using a BlackBerry Enterprise Services (BES) management system and the BlackBerry World – Work application on a BlackBerry 10.3 mobile device platform. The TOE application software does not contain software to update or modify itself. Instead, new versions of the application software are updated via the BlackBerry World – Work application automatically, which removes the previous version and installs a new version, while maintaining access to platform resources. The TOE software is provided by the BES as a ".bar" file. This ".bar" file is digitally signed using a BlackBerry Developer X.509 Certificate. The ".bar" file is cryptographically signed using the BlackBerry Developer X.509 Certificate with a SHA-512 keyed-hash, by the BlackBerry 10 Native SDK. The BES management system verifies the signature on the ".bar" file during installation of the TOE application.

  Removal of the TOE software by a BES management system leaves only configuration data, encrypted files, and potentially audit events.

  The TOE software allows a user to query the current version of the TOE by navigating to "My World" in the BlackBerry World – Work application on the BlackBerry platform and selecting the TOE application to reveal the "Version".

6.7 Trusted path/channels

The Trusted path/channels function is designed to satisfy the following security functional requirements:

- FTP_DIT_EXT.1: The TOE application does not transmit any sensitive data over the network. The only communication occurring as a result of TOE activity is the network traffic associated with checking for the currently available version of the TOE.

7. Platform API List

The following is a list of the platform APIs invoked by the TOE related to supporting the security functionality of the TOE.

- BlackBerry 10 Native Core API

Security Builder API	https://developer.blackberry.com/native/reference/core/com.qnx.doc.crypto.lib_ref/topic/manual/intro.html
Certificate Manager Private API	Private API
Credential Manager API	https://developer.blackberry.com/native/reference/core/com.qnx.doc.credential.lib_ref/topic/manual/credential_overview.html
File System Relay API	Private API
Data Lock Queue Relay API	Private API
System Integrity Services API	Private API
BlackBerry Platform Services API	https://developer.blackberry.com/native/documentation/device_platform/bps/

- BlackBerry 10 Native Cascades API

bb::system::SystemPrompt API	https://developer.blackberry.com/native/reference/cascades/bb__system__systemprompt.html
bb::system::SystemDialog API	https://developer.blackberry.com/native/reference/cascades/bb__system__systemdialog.html
bb::platform::AdarpDomain API	https://developer.blackberry.com/native/reference/cascades/bb__platform__adarpdomain.html
bb::platform::HomeScreen API	http://developer.blackberry.com/native/reference/cascades/platform_home_screen.html
bb::PpsObject API	https://developer.blackberry.com/native/reference/cascades/bb__ppsobject.html

- Qt Core API (http://developer.blackberry.com/native/reference/cascades/qtcore.html addresses all API in this group)
  - QCoreApplication API
  - QSettings API
  - QThread API
  - QTimer API
  - QMutex API
  - QMutexLocker API