Mobile Device Fundamentals

Version: 4.0
2025-01-31
National Information Assurance Partnership

Revision History

Version	Date	Comment
1.0	2013-10-21	Initial Release
1.1	2014-01-12	Typographical changes and additional clarifications in application notes. Removed assignment from FCS_TLS_EXT.1 and limited testing to those ciphersuites in both FCS_TLS_EXT.1 and FCS_TLS_EXT.2.
2.0	2015-09-14	Included changes based on Technical Rapid Response Team Decisions. Clarified many requirements and evaluation activities. Mandated objective requirements: • Application Access Control (FDP_ACF_EXT.1.2) • VPN Information Flow Control (FDP_IFC_EXT.1) Added new objective requirements: • Suite B cryptography for IEEE 802.11 • Certificate enrollment • Protection of additional key material types • Heap overflow protection • Bluetooth requirements • Cryptographic operation services for applications • Remote Attestation (FPT_NOT_EXT.1) Added transition dates for some objective requirements. Included hardware-isolated REK and key storage selections. Allowed key derivation by REK. Clarified FTP_ITC_EXT.1 and added FDP_UPC_EXT.1. Mandated HTTPS and TLS for application use. (FDP_UPC_EXT.1) Removed Dual_EC_DRBG as an approved DRBG. Adopted new TLS requirements. Mandated TSF Wipe upon authentication failure limit and required number of authentication failures be maintained across reboot. Clarified Management Class. Included more domain isolation discussion and tests. Updated Audit requirements and added Auditable Events table. Added SFR Category Mapping Table. Updated Use Case Templates. Moved Glossary to Introduction.
3.0	2015-09-17	Included changes based on Technical Rapid Response Team Decisions. Clarified many requirements and evaluation activities. Mandated objective requirements: • Generation of Audit Records (FAU_GEN.1) • Audit Storage Protection (FAU_STG.1) • Audit Storage Overwrite (FAU_STG.4) • Lock Screen DAR (FDP_DAR_EXT.2) • Discard Bluetooth Connection Attempts from Bluetooth Addresses with Existing Connection (FIA_BLT_EXT.3) • JTAG Disablement (FPT_JTA) Added new objective requirements: • Application Backup • Biometric Authentication Factor • Access Control • User Authentication • Bluetooth Encryption WLAN client requirements moved to Extended Package for WLAN Client. Added SFRs to support BYOD Use Case BYOD Use Case Updated key destruction SFR
3.1	2017-04-05	Included changes based on Technical Rapid Response Team Decisions and incorporated Technical Decisions. Modified biometric requirements: • FIA_UAU.5 - Added iris, face, voice and vein as supported modalities, in addition to fingerprint (allowed in version 3) • FIA_BMG_EXT.1.1 - Clarified AA to specify that vendor evidence is acceptable and expectations of evidence provided. • FIA_BMG_EXT.1.2 - SAFAR was changed to an assignment of a SAFAR no greater than 1:500. • FIA_AFL_EXT.1 - Updated to allow each biometric modality to utilize an individual or shared counter. FCS_TLSC_EXT.1.1 - Removed TLS ciphersuites that utilized SHA1 and updated optional ciphersuites to be uniformed across PPs. FCS_STG_EXT.2.2 - Modified to require long term trusted channel key material be encrypted by an approved method. FIA_UAU_EXT.1.1 - Modified to allow the long term trusted channel key material to be available prior to password being entered at start-up.

3.2	2021-04-15	Removed TLS SFRs and used TLS Functional Package Removed Bluetooth SFRs and utilized Bluetooth Module. Bluetooth SFR moved to Implementation Dependent. FPT_TUD_EXT.4.2 renumbered to FPT_TUD_EXT.5.1
3.3	2022-09-12	Integrated Biometrics cPP Module, Included changes based on Technical Rapid Response Team Decisions and open issues from GitHub. • Removed biometric definitions from Tech Terms • Removed FDP_PBA • Removed FIA_BMG • Updated FIA_UAU.5 to support bio cPP module • Moved FTA_TAB.1 to mandatory • Moved FAU_SAR.1 to mandatory • Added ECD • Updated WLAN Client reference from Extended Package to Module • Removed Diffie-Hellman group 14 selection from FCS_CKM.1.1 and FCS_CKM_EXT.7.1/UNLOCKED
4.0	2024-09-20	• Updated to conform to CC:2022. • Updated TLS package references. • Incorporated X.509 package. • Incorporated applicable technical decisions. • Updated cryptographic algorithm strengths to conform to CNSA.

Contents

1 Introduction

1.1 Overview

The scope of this Protection Profile (PP) is to describe the security functionality of mobile devices in terms of [CC] and to define functional and assurance requirements for such devices.

1.2 Terms

The following sections list Common Criteria and technology terms used in this document.

1.2.1 Common Criteria Terms

Assurance	Grounds for confidence that a TOE meets the SFRs [CC].
Base Protection Profile (Base-PP)	Protection Profile used as a basis to build a PP-Configuration.
Collaborative Protection Profile (cPP)	A Protection Profile developed by international technical communities and approved by multiple schemes.
Common Criteria (CC)	Common Criteria for Information Technology Security Evaluation (International Standard ISO/IEC 15408).
Common Criteria Testing Laboratory	Within the context of the Common Criteria Evaluation and Validation Scheme (CCEVS), an IT security evaluation facility accredited by the National Voluntary Laboratory Accreditation Program (NVLAP) and approved by the NIAP Validation Body to conduct Common Criteria-based evaluations.
Common Evaluation Methodology (CEM)	Common Evaluation Methodology for Information Technology Security Evaluation.
Distributed TOE	A TOE composed of multiple components operating as a logical whole.
Extended Package (EP)	A deprecated document form for collecting SFRs that implement a particular protocol, technology, or functionality. See Functional Packages.
Functional Package (FP)	A document that collects SFRs for a particular protocol, technology, or functionality.
Operational Environment (OE)	Hardware and software that are outside the TOE boundary that support the TOE functionality and security policy.
Protection Profile (PP)	An implementation-independent set of security requirements for a category of products.
Protection Profile Configuration (PP-Configuration)	A comprehensive set of security requirements for a product type that consists of at least one Base-PP and at least one PP-Module.
Protection Profile Module (PP-Module)	An implementation-independent statement of security needs for a TOE type complementary to one or more Base-PPs.
Security Assurance Requirement (SAR)	A requirement to assure the security of the TOE.
Security Functional Requirement (SFR)	A requirement for security enforcement by the TOE.
Security Target (ST)	A set of implementation-dependent security requirements for a specific product.
Target of Evaluation (TOE)	The product under evaluation.

TOE Security Functionality (TSF)	The security functionality of the product under evaluation.
TOE Summary Specification (TSS)	A description of how a TOE satisfies the SFRs in an ST.

1.2.2 Technical Terms

Address Space Layout Randomization (ASLR)	An anti-exploitation feature, which loads memory mappings into unpredictable locations. ASLR makes it more difficult for an attacker to redirect control to code that they have introduced into the address space of a process or the kernel.
Administrator	The Administrator is responsible for management activities, including setting the policy that is applied by the enterprise on the Mobile Device. This administrator is likely to be acting remotely and could be the Mobile Device Management (MDM) Administrator acting through an MDM Agent. If the device is unenrolled, the user is the administrator.
Auxiliary Boot Modes	Auxiliary boot modes are states in which the device provides power to one or more components to provide an interface that enables an unauthenticated user to interact with either a specific component or several components that exist outside of the device’s fully authenticated, operational state.
Biometric Authentication Factor (BAF)	Authentication factor, which uses biometric sample, matched to a biometric authentication template to help establish identity.
Common Application Developer	Application developers (or software companies) often produce many applications under the same name. Mobile devices often allow shared resources by such applications where otherwise resources would not be shared.
Critical Security Parameter (CSP)	Security-related information whose disclosure or modification can compromise the security of a cryptographic module or authentication system.
Data	Program/application or data files that are stored or transmitted by a server or Mobile Device (MD).
Data Encryption Key (DEK)	A key used to encrypt data-at-rest.
Developer Modes	Developer modes are states in which additional services are available to a user in order to provide enhanced system access for debugging of software.
Encrypted Software Keys	These keys are stored in the main file system encrypted by another key and can be changed and sanitized.
Enrolled State	The state in which the Mobile Device is managed with active policy settings from the administrator.
Enterprise Data	Enterprise data is any data residing in the enterprise servers, or temporarily stored on Mobile Devices to which the Mobile Device user is allowed access according to security policy defined by the enterprise and implemented by the administrator.
Ephemeral Keys	These keys are stored in volatile memory.
File Encryption Key (FEK)	A DEK used to encrypt a file or a director when File Encryption is used. FEKs are unique to each encrypted file or directory.
Hardware-Isolated Keys	The OS can only access these keys by reference, if at all, during runtime.
Hybrid Authentication	A hybrid authentication factor is one where a user has to submit a combination of a biometric sample and a PIN or password and both must pass. If either factor fails, the entire attempt fails. The user shall not be made aware of which factor failed, if either fails.
Immutable Hardware Key	These keys are stored as hardware-protected raw key and cannot be changed or sanitized.
Key Chaining	The method of using multiple layers of encryption keys to protect data. A top layer key encrypts a lower layer key, which encrypts the data; this method can have any number of layers.
Key Encryption Key (KEK)	A key used to encrypt other keys, such as DEKs or storage that contains keys.
Locked State	Powered on but most functionality is unavailable for use. User authentication is required to

	access functionality.
MDM Agent	The MDM Agent is installed on a Mobile Device as an application or is part of the Mobile Device's OS. The MDM Agent establishes a secure connection back to the MDM Server controlled by the administrator.
Minutia Point	Friction ridge characteristics that are used to individualize a fingerprint image. Minutia are the points where friction ridges begin, terminate, or split into two or more ridges. In many fingerprint systems, the minutia points are compared for recognition purposes.
Mobile Device (MD)	A device which is composed of a hardware platform and its system software. The device typically provides wireless connectivity and may include software for functions like secure messaging, email, web, VPN (Virtual Private Network) connection, and VoIP (Voice over IP), for access to the protected enterprise network, enterprise data and applications, and for communicating to other Mobile Devices.
Mobile Device Management (MDM)	Mobile device management (MDM) products allow enterprises to apply security policies to mobile devices. This system consists of two primary components: the MDM Server and the MDM Agent.
Mobile Device User	The individual authorized to physically control and operate the Mobile Device. Depending on the use case, this can be the device owner or an individual authorized by the device owner.
Modality	A type or class of biometric system, such as fingerprint recognition, facial recognition, iris recognition, voice recognition, signature/sign, and others.
Mutable Hardware Key	These keys are stored as hardware-protected raw key and can be changed or sanitized.
Operating System (OS)	Software that runs at the highest privilege level and can directly control hardware resources. Modern Mobile Devices typically have at least two primary operating systems: one, which runs on the application processor and one, which runs on the cellular baseband processor. The OS of the application processor handles most user interactions and provides the execution environment for apps. The OS of the cellular baseband processor handles communications with the cellular network and may control other peripherals. The term OS, without context, may be assumed to refer to the OS of the application processor.
PIN Authentication Factor	A PIN is a set of numeric or alphabetic characters that may be used in addition to a biometric factor to provide a hybrid authentication factor. At this time it is not considered as a stand-alone authentication mechanism. A PIN is distinct from a password in that the allowed character set and required length of a PIN is typically smaller than that of a password as it is designed to be input quickly.
Password Authentication Factor	A type of authentication factor requiring the user to provide a secret set of characters to gain access.
Powered Off State	The device has been shut down such that no TOE function can be performed.
Protected Data (PD)	Protected data is all non-TSF data, including all user or enterprise data. Some or all of this data may be considered sensitive data as well.
Root Encryption Key (REK)	A key tied to the device used to encrypt other keys.
Sensitive data	Sensitive data shall be identified in the TSS section of the Security Target (ST) by the ST author. Sensitive data is a subset or all of the Protected data. Sensitive data may include all user or enterprise data or may be specific application data such as emails, messaging, documents, calendar items, and contacts. Sensitive data is protected while in the locked state (FDP_DAR_EXT.2).
Software Keys	The OS access the raw bytes of these keys during runtime.
TSF Data	Data for the operation of the TSF upon which the enforcement of the requirements relies.
Trust Anchor Database	A list of trusted root Certificate Authority certificates.
Unenrolled State	The state in which the Mobile Device is not managed.
Unlocked State	Powered on and device functionality is available for use. Implies user authentication has occurred (when so configured).
Verification	A task where the biometric system attempts to confirm an individual's claimed identity by comparing a submitted sample to one or more previously enrolled authentication templates.

1.3 TOE Overview

This assurance standard specifies information security requirements for Mobile Devices for use in an enterprise. A Mobile Device in the context of this assurance standard is a device, which is composed of a hardware platform and its system software. The device typically provides wireless connectivity and may include software for functions like secure messaging, email, web, VPN connection, and VoIP (Voice over IP), for access to the protected enterprise network, enterprise data and applications, and for communicating to other Mobile Devices.

Figure 1 illustrates the network operating environment of the Mobile Device.

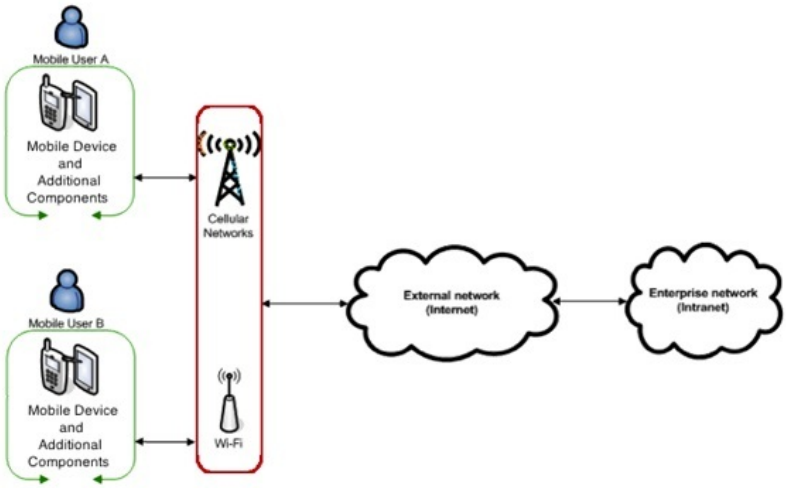


Figure 1: Mobile Device Network Environment

Examples of a "Mobile Device" that should claim conformance to this Protection Profile include smartphones, tablet computers, and other Mobile Devices with similar capabilities.

The Mobile Device provides essential services, such as cryptographic services, data-at-rest protection, and key storage services to support the secure operation of applications on the device. Additional security features such as security policy enforcement, application mandatory access control, anti-exploitation features, user authentication, and software integrity protection are implemented in order to address threats.

This assurance standard describes these essential security services provided by the Mobile Device and serves as a foundation for a secure mobile architecture. The wireless connectivity shall be validated against the PP-Module for Wireless LAN Clients, version 2.0. If the mobile device contains Bluetooth functionality (i.e., has Bluetooth hardware), the Bluetooth connectivity shall be evaluated against the PP-Module for Bluetooth, version 2.0. As illustrated in Figure 2, it is expected that a typical deployment would also include either third-party or bundled components. Whether these components are bundled as part of the Mobile Device by the manufacturer or developed by a third-party, they must be separately validated against the related assurance standards such as the PP-Module for MDM Agent, PP-Module for VPN Client, PP-Module for VVoIP, and cPP-Module for Biometrics. It is the responsibility of the architect of the overall secure mobile architecture to ensure validation of these components. Additional applications that may come pre-installed on the Mobile Device that are not validated are considered to be potentially flawed, but not malicious. Examples include email client and web browser.

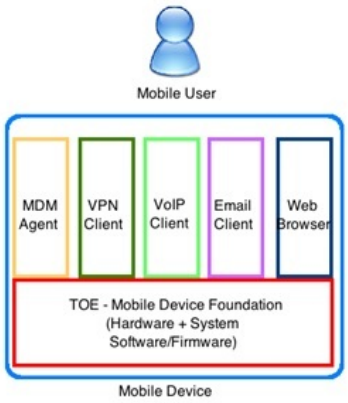


Figure 2: Optional Additional Mobile Device Components

1.4 Use Cases

The Mobile Device may be operated in a number of use cases. use-case-appendix provides use case templates that list those selections, assignments, and objective requirements that best support the use cases identified by this Protection Profile. In addition to providing essential security services, the Mobile Device includes the necessary security functionality to support configurations for these various use cases. Each use case may require additional configuration and applications to achieve the desired security. A selection of these use

cases is elaborated below.

Several of the use case templates include objective requirements that are strongly desired for the indicated use cases. Readers can expect those requirements to be made mandatory in a future revision of this protection profile, and industry should aim to include that security functionality in products in the near-term.

As of publication of this version of the Protection Profile, meeting the requirements in is necessary for all use cases.

[USE CASE 1] Enterprise-owned device for general-purpose enterprise use and limited personal use

> An enterprise-owned device for general-purpose business use is commonly called *Corporately Owned, Personally Enabled (COPE)*. This use case entails a significant degree of enterprise control over configuration and, possibly, software inventory. The enterprise elects to provide users with Mobile Devices and additional applications (such as VPN or email clients) in order to maintain control of their Enterprise data and security of their networks. Users may use Internet connectivity to browse the web or access corporate mail or run enterprise applications, but this connectivity may be under significant control of the enterprise.

[USE CASE 2] Enterprise-owned device for specialized, high security use

> An enterprise-owned device with intentionally limited network connectivity, tightly controlled configuration, and limited software inventory is appropriate for specialized, high-security use cases. For example, the device may not be permitted connectivity to any external peripherals. It may only be able to communicate via its Wi-Fi or cellular radios with the enterprise-run network, which may not even permit connectivity to the Internet. Use of the device may entail compliance with policies that are more restrictive than those in any general-purpose use case, yet may mitigate risks to highly sensitive information. As in the previous case, the enterprise will look for additional applications providing enterprise connectivity and services to have a similar level of assurance as the platform.

[USE CASE 3] Personally-owned device for personal and enterprise use

> A personally-owned device that is used for both personal activities and enterprise data is commonly called Bring Your Own Device (BYOD). The device may be provisioned for access to enterprise resources after significant personal usage has occurred. Unlike in the enterprise-owned cases, the enterprise is limited in what security policies it can enforce because the user purchased the device primarily for personal use and is unlikely to accept policies that limit the functionality of the device. However, because the enterprise allows the user full (or nearly full) access to the enterprise network, the enterprise will require their own security controls to ensure that enterprise resources are protected from potential threats posed by the personal activities on the device. These controls could potentially be enforced by a separation mechanism built-in to the device itself to distinguish between enterprise and personal activities, or by a third-party application that provides access to enterprise resources and leverages security capabilities provided by the mobile device. Based upon the operational environment and the acceptable risk level of the enterprise, those security functional requirements outlined in of this PP along with the selections in the Use Case 3 template defined in Appendix F - Use Case Templates are sufficient for the secure implementation of this BYOD use case.

[USE CASE 4] Personally-owned device for personal and limited enterprise use

> A personally-owned device that is used for both personal activities and enterprise data is commonly called Bring Your Own Device (BYOD). This device may be provisioned for limited access to enterprise resources such as enterprise email. Because the user does not have full access to the enterprise or enterprise data, the enterprise may not need to enforce any security policies on the device. However, the enterprise may want secure email and web browsing with assurance that the services being provided to those clients by the Mobile Device are not compromised. Based upon the operational environment and the acceptable risk level of the enterprise, those security functional requirements outlined in Section 5 Security Requirements of this PP are sufficient for the secure implementation of this BYOD use case.

1.5 Package Usage

This section contains selections and assignments that are required when the listed Functional Packages are claimed by this PP. To support FTP_ITC_EXT.1 and FDP_UPC_EXT.1/APPS, the ST author shall claim support for TLS as a client. If DTLS is claimed in FTP_ITC_EXT.1, the ST author shall also claim support for DTLS as a client. Support for TLS as a server or DTLS as a server shall not be claimed. To support FTP_ITC_EXT.1 and FDP_UPC_EXT.1/APPS, the ST author shall claim support for mutual authentication in FCS_TLSC_EXT.1.1. To support FCS_HTTPS_EXT.1.3, the ST author shall claim how the TOE handles invalid certificates in a manner that is consistent with the claims made in FCS_HTTPS_EXT.1.3. Regardless of the claims made for FCS_HTTPS_EXT.1.3, to support FDP_UPC_EXT.1/APPS, the ST author shall claim in FCS_TLSC_EXT.1.6 that invalid certificates are rejected with no exceptions in this specific case. Because support for TLS client mutual authentication is required, the ST author shall claim FCS_TLSC_EXT.2. To support FTP_ITC_EXT.1 and FDP_UPC_EXT.1/APPS specifically in the case where DTLS is claimed in FTP_ITC_EXT.1.1, the ST author shall claim support for mutual authentication in FCS_DTLSC_EXT.1.1. DTLS support is not mandatory. However, if DTLS is claimed in FTP_ITC_EXT.1 or FDP_UPC_EXT.1/APPS, it is necessary for the TOE to support mutual authentication as a DTLS client. In this case, FCS_DTLS_EXT.2 shall be claimed as a result of the selections made in FCS_DTLSC_EXT.1.1. The ST author shall select the options to verify and assert certificate identities. Because the TOE is the entire mobile device and encompasses the platform, the ST author shall not claim platform-provided functionality. In other words, any applicable SFRs shall select the option to implement or provide the functionality, rather than rely on a platform. These may include one or more of the following, depending on the TOE, but apply to all package requirements that have such a selection:

- FIA_CMCC_EXT.1
- FIA_CMPC_EXT.1

- FIA_ESTC_EXT.1
- FIA_X509_EXT.1
- FIA_X509_EXT.2
- FIA_X509_EXT.3

The ST author shall select the option to request a certificate from an external CA in FIA_XCU_EXT.2.1 and shall not select any options elsewhere in the package that involve claiming the ability to be a CA. The TOE must utilize appropriate cryptographic algorithms that conform to CNSA standards. Thus, the ST author shall select to utilize no other algorithms outside of those specified in RFC 8603 for certificate or CRL signatures. Additionally, the ST author shall not select to use ECDSA with SHA-512 signatures for OCSP responses, and shall select to utilize no other algorithms for OCSP responses. The ST author shall select the options to process the basicConstraints and extendedKeyUsage extensions. Other extensions may be selected as appropriate without restriction. The ST author shall select only from options involving CRL or OCSP in FIA_X509_EXT.1.3. The ST author shall not select the options to not obtain revocation status information at any time, nor to utilize local revocation information or utilize a network connection to the CA to obtain its revocation information. Revocation information must be retrieved via an option involving CRL or OCSP. The ST author shall not select the option to pass certificate information to supported functions and shall select the option to verify certificates by evaluating the extendedKeyUsage field in the leaf certificate. Additionally, the ST author shall not select the option to validate certificates against any rules for S/MIME certificates. The ST author shall not select options that involve the S/MIME or SSH protocols. If the TOE supports TSF integrity testing (i.e., FPT_TST_EXT.3 is claimed), FIA_X509_EXT.2.1 must include a claim that X.509 certificates are supported for code signing for software integrity testing. If the TOE supports trusted update verification using code signing certificates (i.e., FPT_TST_EXT.4 is claimed in the context of the TOE's own updates), FIA_X509_EXT.2.1 must include a claim that X.509 certificates are supported for code signing for system software updates. If the TOE supports code signing for mobile apps (i.e., FPT_TUD_EXT.4 is claimed in the context of external apps), the assignment in FIA_X509_EXT.2.1 must be completed to reference "code signing for mobile applications." The ST author shall not select options that involve the S/MIME or SSH protocols.

1.6 Product Features Mapped to Implementation-dependent Requirements

The feature(s) enumerated below, if implemented by the TOE, require that additional SFRs be claimed in the ST.

1.6.1 Bluetooth

Bluetooth is a short-range wireless technology standard that is commonly used for exchanging data between devices over short distances. Most, if not all, mobile devices include Bluetooth hardware.

If this feature is implemented by the TOE, the following requirements must be claimed in the ST:

- **FDP_UPC_EXT.1/BLUETOOTH**

1.6.2

If this feature is implemented by the TOE, the following requirements must be claimed in the ST:

1.6.3

If this feature is implemented by the TOE, the following requirements must be claimed in the ST:

2 Conformance Claims

Conformance Statement

An ST must claim exact conformance to this PP.

The evaluation methods used for evaluating the TOE are a combination of the workunits defined in [CEM] as well as the Evaluation Activities for ensuring that individual SFRs and SARs have a sufficient level of supporting evidence in the Security Target and guidance documentation and have been sufficiently tested by the laboratory as part of completing ATE_IND.1. Any functional packages this PP claims similarly contain their own Evaluation Activities that are used in this same manner.

CC Conformance Claims

This PP is conformant to Part 2 (extended) and Part 3 (extended) of Common Criteria CC:2022, Revision 1.

PP Claim

This PP does not claim conformance to any Protection Profile.

The following PPs and PP-Modules are allowed to be specified in a PP-Configuration with this PP:

- PP-Module for Virtual Private Network (VPN) Clients, version 3.0
- PP-Module for Bluetooth, version 2.0
- PP-Module for Mobile Device Management Agent, version 2.0
- PP-Module for Wireless LAN Clients, version 2.0
- cPP-Module for Biometric Enrolment and Verification, version 1.1

Package Claim

- This PP is Functional Package for Transport Layer Security Version 2.1 conformant.
- This PP is Functional Package for X.509 Version 1.0 conformant.
- This PP does not conform to any assurance packages.

The functional packages to which the PP conforms may include SFRs that are not mandatory to claim for the sake of conformance. An ST that claims one or more of these functional packages may include any non-mandatory SFRs that are appropriate to claim based on the capabilities of the TSF and on any triggers for their inclusion based inherently on the SFR selections made.

3 Security Problem Definition

3.1 Threats

Mobile devices are subject to the threats of traditional computer systems along with those entailed by their mobile nature. The threats considered in this PP are those of network eavesdropping, network attacks, physical access, malicious or flawed applications, persistent presence, and backup as detailed in the following sections.

T.MALICIOUS_APP
Applications loaded onto the Mobile Device may include malicious or exploitable code. This code could be included intentionally or unknowingly by the developer, perhaps as part of a software library. Malicious apps may attempt to exfiltrate data to which they have access. They may also conduct attacks against the platform’s system software, which will provide them with additional privileges and the ability to conduct further malicious activities. Malicious applications may be able to control the device's sensors (GPS, camera, microphone) to gather intelligence about the user's surroundings even when those activities do not involve data resident or transmitted from the device. Flawed applications may give an attacker access to perform network-based or physical attacks that otherwise would have been prevented.

T.NETWORK_ATTACK
An attacker is positioned on a wireless communications channel or elsewhere on the network infrastructure. Attackers may initiate communications with the Mobile Device or alter communications between the Mobile Device and other endpoints in order to compromise the Mobile Device. These attacks include malicious software update of any applications or system software on the device. These attacks also include malicious web pages or email attachments, which are usually delivered to devices over the network.

T.NETWORK_EAVESDROP
An attacker is positioned on a wireless communications channel or elsewhere on the network infrastructure. Attackers may monitor and gain access to data exchanged between the Mobile Device and other endpoints, resulting in modification or disclosure of sensitive communications.

T.PERSISTENT_PRESENCE
Persistent presence on a device by an attacker implies that the device has lost integrity and cannot regain it. The device has likely lost this integrity due to some other threat vector, yet the continued access by an attacker constitutes an on-going threat in itself. In this case, the device and its data may be controlled by an adversary as well as by its legitimate owner.

T.PHYSICAL_ACCESS
An attacker, with physical access, may attempt to access user data on the Mobile Device including credentials. These physical access threats may involve attacks, which attempt to access the device through external hardware ports, impersonate the user authentication mechanisms, through its user interface, and also through direct and possibly destructive access to its storage media. Note: Defending against device re-use after physical compromise is out of scope for this Protection Profile.

3.2 Assumptions

The specific conditions listed below are assumed to exist in the TOE’s Operational Environment. These include both practical realities in the development of the TOE security requirements and the essential environmental conditions on the use of the TOE.

A.CONFIG
It is assumed that the TOE’s security functions are configured correctly in a manner to ensure that the TOE security policies will be enforced on all applicable network traffic flowing among the attached networks.

A.NOTIFY
It is assumed that the mobile user will immediately notify the administrator if the Mobile Device is lost or stolen.

A.PRECAUTION
It is assumed that the mobile user exercises precautions to reduce the risk of loss or theft of the Mobile Device.

A.PROPER_USER
Mobile Device users are not willfully negligent or hostile, and use the device within compliance of a reasonable Enterprise security policy.

3.3 Organizational Security Policies

This document does not define any additional OSPs.

4 Security Objectives

4.1 Security Objectives for the Operational Environment

The following security objectives for the operational environment assist the OS in correctly providing its security functionality. These track with the assumptions about the environment.

OE.CONFIG
TOE administrators will configure the Mobile Device security functions correctly to create the intended security policy

OE.DATA_PROPER_USER
Administrators take measures to ensure that mobile device users are adequately vetted against malicious intent and are made aware of the expectations for appropriate use of the device.

OE.NOTIFY
The Mobile User will immediately notify the administrator if the Mobile Device is lost or stolen.

OE.PRECAUTION
The mobile device user exercises precautions to reduce the risk of loss or theft of the Mobile Device.

4.2 Security Objectives Rationale

This section describes how the assumptions and organizational security policies map to operational environment security objectives.

Table 1: Security Objectives Rationale

Assumption or OSP	Security Objectives	Rationale
A.CONFIG	OE.CONFIG	The operational environment objective OE.CONFIG is realized through A.CONFIG.
A.NOTIFY	OE.NOTIFY	The operational environment objective OE.NOTIFY is realized through A.NOTIFY.
A.PRECAUTION	OE.PRECAUTION	The operational environment objective OE.PRECAUTION is realized through A.PRECAUTION.
A.PROPER_USER	OE.DATA_PROPER_USER	The operational environment objective OE.DATA_PROPER_USER is realized through A.PROPER_USER.

5 Security Requirements

This chapter describes the security requirements which have to be fulfilled by the product under evaluation. Those requirements comprise functional components from Part 2 and assurance components from Part 3 of [CC]. The following conventions are used for the completion of operations:

- **Refinement** operation (denoted by **bold text** or ~~strikethrough text~~): Is used to add details to a requirement or to remove part of the requirement that is made irrelevant through the completion of another operation, and thus further restricts a requirement.
- **Selection** (denoted by *italicized text*): Is used to select one or more options provided by the [CC] in stating a requirement.
- **Assignment** operation (denoted by *italicized text*): Is used to assign a specific value to an unspecified parameter, such as the length of a password. Showing the value in square brackets indicates assignment.
- **Iteration** operation: Is indicated by appending the SFR name with a slash and unique identifier suggesting the purpose of the operation, e.g. "/EXAMPLE1."

5.1 Security Functional Requirements

5.1.1 Auditable Events for Mandatory SFRs

Table 2: Auditable Events for Mandatory Requirements

Requirement	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	No events specified	N/A
FAU_SAR.1	No events specified	N/A
FAU_STG.2	No events specified	N/A
FAU_STG.5	No events specified	N/A
FCS_CKM.1/AKG	No events specified	N/A
FCS_CKM.1/SKG	No events specified	N/A
FCS_CKM.2	No events specified	N/A
FCS_CKM.6	No events specified	N/A
FCS_CKM_EXT.1	[**selection, choose one of**: *generation of a REK*, *none*]	No additional information
FCS_CKM_EXT.2	No events specified	N/A
FCS_CKM_EXT.3	No events specified	N/A
FCS_CKM_EXT.5	[**selection, choose one of**: *Failure of the wipe*, *none*]	No additional information
FCS_CKM_EXT.6	No events specified	N/A
FCS_CKM_EXT.7/LOCKED	No events specified	N/A
FCS_CKM_EXT.8	No events specified	N/A
FCS_COP.1/AEAD	No events specified	N/A
FCS_COP.1/Hash	No events specified	N/A
FCS_COP.1/KeyWrap	No events specified	N/A
FCS_COP.1/KeyedHash	No events specified	N/A
FCS_COP.1/SigGen	No events specified	N/A
FCS_COP.1/SigVer	No events specified	N/A
FCS_HTTPS_EXT.1	Failure of the certificate validity check.	• Issuer Name and Subject Name of certificate • [**selection, choose one of**: *User's authorization decision*, *No additional information*]
FCS_IV_EXT.1	No events specified	N/A
FCS_RBG.1	Failure of the randomization process	None.
FCS_RBG.6	No events specified	N/A

FCS_SRV_EXT.1	No events specified	N/A
FCS_STG_EXT.1	[**selection, choose one of**: *Exceptions to use and destruction rules*, *none*]	Identity of key, role and identity of requester
	Import or destruction of key	Identity of key, role and identity of requester
FCS_STG_EXT.2	No events specified	N/A
FCS_STG_EXT.3	Failure to verify integrity of stored key	Identity of key being verified
FDP_ACF_EXT.1	No events specified	N/A
FDP_DAR_EXT.1	[**selection, choose one of**: *Failure to encrypt/decrypt data*, *none*]	No additional information
FDP_DAR_EXT.2	[**selection, choose one of**: *Failure to encrypt/decrypt data*, *none*]	No additional information
FDP_IFC_EXT.1	No events specified	N/A
FDP_STG_EXT.1	Addition or removal of certificate from Trust Anchor Database	Subject name of certificate.
FDP_UPC_EXT.1/APPS	Application initiation of trusted channel	Name of application. Trusted channel protocol. Non-TOE endpoint of connection
FIA_AFL_EXT.1	Excess of authentication failure limit	Authentication factor used
FIA_PMG_EXT.1	No events specified	N/A
FIA_TRT_EXT.1	No events specified	N/A
FIA_UAU.5	No events specified	N/A
FIA_UAU.6/CREDENTIAL	No events specified	N/A
FIA_UAU.6/LOCKED	User changes Password Authentication Factor	No additional information
FIA_UAU.7	No events specified	N/A
FIA_UAU_EXT.1	No events specified	N/A
FIA_UAU_EXT.2	Action performed before authentication.	No additional information
FIA_X509_EXT.6	No events specified	N/A
FMT_MOF_EXT.1	No events specified	N/A
FMT_SMF_EXT.1	Initiation of policy update	Policy name
	Change of settings	Role of user that changed setting, Value of new setting
	Success or failure of function	Role of user that performed function, Function performed, Reason for failure
	Initiation of software update	Version of update
	Initiation of application installation or update	Name and version of application
FMT_SMF_EXT.2	Unenrollment, Initiation of unenrollment	Identity of administrator Remediation action performed, failure of accepting command to unenroll
FPT_AEX_EXT.1	No events specified	N/A
FPT_AEX_EXT.2	No events specified	N/A
FPT_AEX_EXT.3	No events specified	N/A
FPT_AEX_EXT.4	No events specified	N/A
FPT_FLS.1	No events specified	N/A

FPT_JTA_EXT.1	No events specified	N/A
FPT_KST_EXT.1	No events specified	N/A
FPT_KST_EXT.2	No events specified	N/A
FPT_KST_EXT.3	No events specified	N/A
FPT_NOT_EXT.1	[**selection, choose one of**: *Measurement of TSF software*, *none*]	[**selection, choose one of**: *Integrity verification value*, *No additional information*]
FPT_STM.1	No events specified	N/A
FPT_TST.1	Execution of self-tests.	None.
FPT_TST_EXT.1	Initiation of self-test	No additional information
	Failure of self-test	[**selection, choose one of**: *Algorithm that caused the failure*, *No additional information*]
FPT_TST_EXT.2/PREKERNEL	Start-up of TOE	No additional information
	[**selection, choose one of**: *Detected integrity violation*, *none*]	[**selection, choose one of**: *The TSF code file that caused the integrity violation*, *No additional information*]
FPT_TUD_EXT.1	No events specified	N/A
FPT_TUD_EXT.2	Success or failure of signature verification for software updates	No additional information
FPT_TUD_EXT.3	Success or failure of signature verification for applications	No additional information
FTA_SSL_EXT.1	No events specified	N/A
FTA_TAB.1	No events specified	N/A
FTP_ITC_EXT.1	Initiation and termination of trusted channel	Trusted channel protocol, non-TOE endpoint of connection

5.1.2 Class: Security Audit (FAU)

FAU_GEN.1 Audit Data Generation

FAU_GEN.1.1

The TSF shall be able to generate audit data of the following auditable events:

- Start-up and shutdown of the audit functions
- All auditable events for the [*not specified*] level of audit
  [
- *All administrative actions*
- *Start-up and shutdown of the OS*
- *Insertion or removal of removable media*
- *Specifically defined auditable events in Table 2*
- Auditable events as defined in the Functional Package for Transport Layer Security (TLS), version 2.1;
- Auditable events as defined in the Functional Package for X.509, version 1.0;
- *[**selection**: Audit data reaching [**assignment**: integer value less than 100] percent of audit capacity, Auditable events defined in Table t-audit-objective for Objective SFRs, Auditable events defined in Table t-audit-impl-dep for Implementation-Dependent SFRs, Auditable events defined in Table t-audit-sel-based for Selection-Based SFRs, no additional auditable events]]*

Application Note: Administrator actions are defined as functions labeled as mandatory for FMT_MOF_EXT.1.2 (i.e. 'M-MM' in Table 20). If the TSF does not support removable media, number 4 is implicitly met.

The TSF must generate audit data for all events contained in Table 2. For all other audit tables (i.e., tables for non-mandatory SFRs and references to functional packages), the ST author includes the audit events (if any) for the SFRs that are claimed in the ST.

Table 2 Application Note:
FPT_TST_EXT.1 – Audit of self-tests is required only at initial start-up. Since the TOE "transitions to non-operational mode" upon failure of a self-test, per FPT_NOT_EXT.1, this is considered equivalent evidence to audit data for the failure of a self-test.

FDP_DAR_EXT.1 - "None" must be selected, if the TOE utilizes whole volume encryption for protected data, since it is not feasible to audit when the encryption/decryption fails. If the TOE utilizes file-based encryption for protected data and audits when this encryption/decryption fails, then that auditable event should be selected.

For FMT_SMF_EXT.1, it is acceptable for the initiation of the software update to be audited without indicating the outcome (success or failure) of the update.

FAU_GEN.1.2

The TSF shall record within the audit data at least the following information:

- Date and time of the event
- Type of event
- Subject identity
- The outcome (success or failure) of the event
- **Additional information in Table 2**
- **Additional information defined in the Functional Package for Transport Layer Security (TLS), version 2.1;**
- **Additional information defined in the Functional Package for X.509, version 1.0;**
- **[selection: *Additional information defined in Table t-audit-objective for Objective SFRs, Additional information defined in Table t-audit-impl-dep for Implementation-Dependent SFRs, Additional information defined in Table t-audit-sel-based for Selection-Based SFRs, no additional information*]**

Application Note: The subject identity is usually the process name/ID. The event type is often indicated by a severity level, for example, 'info', 'warning', or 'error'.

For each audit event selected from in FAU_GEN.1.1 if additional information is required to be recorded within the audit data, it should be included in this selection.

Evaluation Activities

FAU_GEN.1
TSS
The evaluator shall check the TSS and ensure that it lists all of the auditable events and provides a format for audit data. Each audit data format type must be covered, along with a brief description of each field. The evaluator shall check to make sure that every audit event type mandated by the PP is described and that the description of the fields contains the information required in FAU_GEN.1.2.

Guidance
The evaluator shall also make a determination of the administrative actions that are relevant in the context of this PP including those listed in the Management section. The evaluator shall examine the administrative guide and make a determination of which administrative commands are related to the configuration (including enabling or disabling) of the mechanisms implemented in the TOE that are necessary to enforce the requirements specified in the PP. The evaluator shall document the methodology or approach taken while determining which actions in the administrative guide are security relevant with respect to this PP. The evaluator may perform this activity as part of the activities associated with ensuring the AGD_OPE guidance satisfies the requirements.

Tests
The evaluator shall test the TOE's ability to correctly generate audit data by having the TOE generate audit data for the events listed in the provided table and administrative actions. This should include all instances of an event. The evaluator shall test that audit data are generated for the establishment and termination of a channel for each of the cryptographic protocols contained in the ST. For administrative actions, the evaluator shall test that each action determined by the evaluator above to be security relevant in the context of this PP is auditable. When verifying the test results, the evaluator shall ensure the audit data generated during testing match the format specified in the administrative guide, and that the fields specified in FAU_GEN.1.2 are contained in the audit data.

Note that the testing here can be accomplished in conjunction with the testing of the security mechanisms directly. For example, testing performed to ensure that the administrative guidance provided is correct verifies that AGD_OPE.1 is satisfied and should address the invocation of the administrative actions that are needed to verify the audit data are generated as expected.

FAU_SAR.1 Audit Review

FAU_SAR.1.1

The TSF shall provide [*the administrator*] with the capability to read [*all audited events and data contents*] from the audit data.

Application Note: The administrator must have access to read the audit data, perhaps through an API or via an MDM Agent, which transfers the local records stored on the TOE to the MDM Server where the enterprise administrator may view them. If this requirement is included in the ST, function 32 must be included in the selection of FMT_SMF_EXT.1.

FAU_SAR.1.2

The TSF shall provide the audit data in a manner suitable for the user to interpret the information.

Evaluation Activities

FAU_SAR.1
TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests
The evaluation activity for this requirement is performed in conjunction with test for function 32 of FMT_SMF_EXT.1.

FAU_SEL.1 Selective Audit

This is an objective component.

FAU_SEL.1.1

The TSF shall be able to select the set of events to be audited from the set of all auditable events based on the following attributes

a. [*event type*]
b. [*success of auditable security events*
c. *failure of auditable security events*
d. *[**assignment**: other attributes]*]

Application Note: The intent of this requirement is to identify all criteria that can be selected to trigger an audit event. This can be configured through an interface on the TSF for a user or administrator to invoke. For the ST author, the assignment is used to list any additional criteria or "none".

Evaluation Activities

FAU_SEL.1
TSS
There are no TSS evaluation activities for this component.

Guidance
The evaluator shall review the administrative guidance to ensure that the guidance itemizes all event types, as well as describes all attributes that are to be selectable in accordance with the requirement, to include those attributes listed in the assignment. The administrative guidance shall also contain instructions on how to set the pre-selection as well as explain the syntax (if present) for multi-value pre-selection. The administrative guidance shall also identify those audit data that are always recorded, regardless of the selection criteria currently being enforced.

Tests
The evaluator shall also perform the following tests:

- *Test FAU_SEL.1:1: For each attribute listed in the requirement, the evaluator shall devise a test to show that selecting the attribute causes only audit events with that attribute (or those that are always recorded, as identified in the administrative guidance) to be recorded.*
- *Test FAU_SEL.1:2: [conditional] If the TSF supports specification of more complex audit pre-selection criteria (e.g., multiple attributes, logical expressions using attributes) then the evaluator shall devise tests showing that this capability is correctly implemented. The evaluator shall also, in the test plan, provide a short narrative justifying the set of tests as*

representative and sufficient to exercise the capability.

FAU_STG.2 Protected Audit Data Storage

FAU_STG.2.1

The TSF shall protect the stored audit data in the audit trail from unauthorized deletion.

FAU_STG.2.2

The TSF shall be able to [*prevent*] unauthorized modifications to the stored audit data in the audit trail.

Evaluation Activities

FAU_STG.2
TSS
The evaluator shall ensure that the TSS lists the location of all logs and the access controls of those files such that unauthorized modification and deletion are prevented.

Guidance
There are no guidance evaluation activities for this component.

Tests
- *Test FAU_STG.2:1: The evaluator shall attempt to delete the audit trail in a manner that the access controls should prevent (as an unauthorized user) and shall verify that the attempt fails.*
- *Test FAU_STG.2:2: The evaluator shall attempt to modify the audit trail in a manner that the access controls should prevent (as an unauthorized application) and shall verify that the attempt fails.*

FAU_STG.5 Prevention of Audit Data Loss

FAU_STG.5.1

The TSF shall [*overwrite the oldest stored audit records*] ~~[assignment: other actions to be taken in case of audit storage failure and conditions for the actions]~~ if the audit data storage is full.

Evaluation Activities

FAU_STG.5
TSS
The evaluator shall examine the TSS to ensure that it describes the size limits on the audit records, the detection of a full audit trail, and the actions taken by the TSF when the audit trail is full. The evaluator shall ensure that the actions results in the deletion or overwrite of the oldest stored record.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

5.1.3 Class: Cryptographic Support (FCS)

This section describes how keys are generated, derived, combined, released and destroyed. There are two major types of keys: DEKs and KEKs. (A REK is considered a KEK.) DEKs are used to protect data (as in the DAR protection described in FDP_DAR_EXT.1 and FDP_DAR_EXT.2). KEKs are used to protect other keys – DEKs, other KEKs, and other types of keys stored by the user or applications. The following diagram shows an example key hierarchy to illustrate the concepts of this profile. This example is not meant as an approved design, but ST authors will be expected to provide a diagram illustrating their key hierarchy in order to demonstrate that they meet the requirements of this profile. Please note if biometric in accordance with the Biometric Enrollment and Verification, version 1.1 is selected in FIA_UAU.5.1, each BAF claimed in FIA_MBV_EXT.1.1 in the Biometric Enrollment and Verification, version 1.1 shall be illustrated in the key hierarchy diagram, to include a description of when and how the BAF is used to release keys. If hybrid is selected in FIA_UAU.5.1, meaning that a PIN or password must be used in conjunction with the BAF, this interaction shall be included.

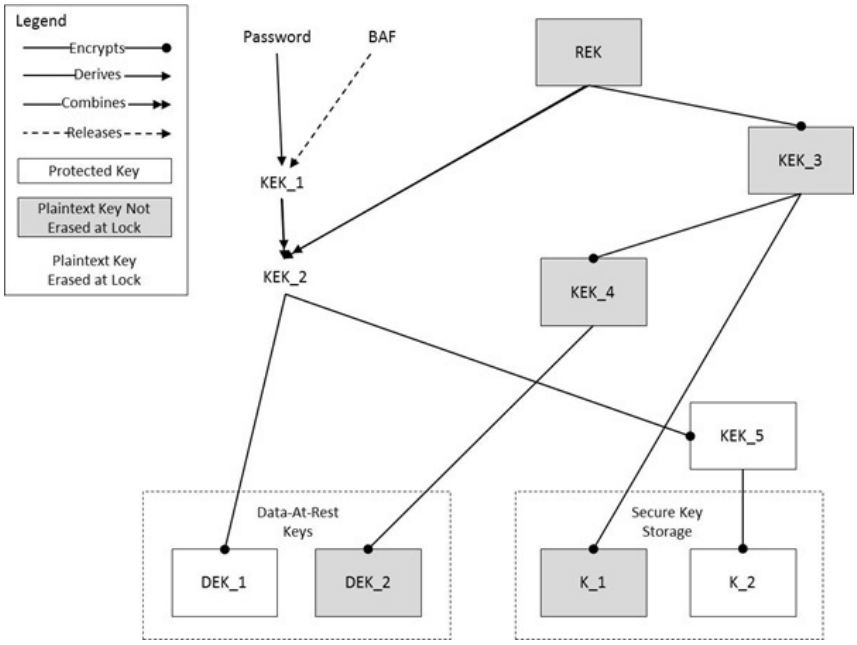


Figure 3: An Illustrative Key Hierarchy

FCS_CKM.1/AKG Cryptographic Key Generation - Asymmetric Key

FCS_CKM.1.1/AKG

The TSF shall generate **asymmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm [**selection**: *Cryptographic Key Generation Algorithm*] and specified cryptographic **algorithm parameters** ~~key sizes~~ [**selection**: *Cryptographic Algorithm Parameters*] that meet the following: [**selection**: *List of Standards*]

The following table provides the allowable choices for completion of the selection operations of FCS_CKM.1/AKG.

Table 3: Allowable Choices for FCS_CKM.1/AKG

Identifier	Cryptographic Key Generation Algorithm	Cryptographic Algorithm Parameters	List of Standards
RSA	RSA	Modulus of size [**selection**: *3072*, *4096*, *6144*, *8192*] bits	NIST FIPS PUB 186-5 (Section A.1.1)
ECC-ERB	ECC-ERB - Extra Random Bits	Elliptic Curve [**selection**: *P-384*, *P-521*]	FIPS PUB 186-5 (Section A.2.1) NIST SP 800-186 (Section 3) [NIST Curves]
ECC-RS	ECC-RS - Rejection Sampling	Elliptic Curve [**selection**: *P-384*, *P-521*]	FIPS PUB 186-5 (Section A.2.2) NIST SP 800-186 (Section 3) [NIST Curves]
FFC-ERB	FFC-ERB - Extra Random Bits	Static domain parameters approved for [**selection**: • *IKE Groups [**selection**: MODP-3072, MODP-4096, MODP-6144, MODP-8192]* • *TLS Groups [**selection**: ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]*]	NIST SP 800-56A Revision 3 (Section 5.6.1.1.3) [key pair generation] [**selection**: *RFC 3526 [IKE groups]*, *RFC 7919 [TLS groups]*]
FFC-RS	FFC-RS - Extra Random Bits	Static domain parameters approved for [**selection**: • *IKE Groups [**selection**: MODP-3072, MODP-4096,*	NIST SP 800-56A Revision 3 (Section 5.6.1.1.3) [key pair generation] [**selection**: *RFC*

		MODP-6144, MODP-8192] • *TLS Groups [**selection**: ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]*]	*3526 [IKE groups], RFC 7919 [TLS groups]*]
LMS	LMS	Private key size =undefined Winternitz parameter =undefined Tree height = [**selection**: *5*, *10*, *15*, *20*, *25*]	RFC 8554 [LMS] NIST SP 800-208 [parameters]
ML-KEM	ML-KEM KeyGen	Parameter set = ML-KEM-1024	NIST FIPS 203 (Section 7.1)
ML-DSA	ML-DSA KeyGen	Parameter set = ML-DSA-87	NIST FIPS 204 (Section 5.1)
XMSS	XMSS	Private key size =undefined Tree height = [**selection**: *10*, *16*, *20*]	RFC 8391 [XMSS] NIST SP 800-208 [parameters]

Application Note: For RSA the choice of the modulus implies the resulting key sizes of the public and private keys generated using the specified standard methods. RSA key generation with modulus size 2048 bits is no longer permitted by CNSA.

For Finite Field Cryptography (FFC) DSA, ST authors should consult schemes for guidelines on use. FIPS PUB 186-5 does not approve DSA for digital signature generation but allows DSA for digital signature verification for legacy purposes. “FFC-ERB” or “FFC–RS” may be claimed only for generating private and public keys when “DH” is claimed in FCS_CKM_EXT.7/LOCKED or FCS_CKM_EXT.7/UNLOCKED.

When generating ECC keys pairs for key agreement and if “ECDH” is claimed in FCS_CKM_EXT.7/LOCKED or FCS_CKM_EXT.7/UNLOCKED, then “ECC–ERB” or “ECC–RS” must be claimed. The sizes of the private key, which is a scalar, and the public key, which is a point on the elliptic curve, are determined by the choice of the curve.

When generating ECC key pairs for digital signature generation and if “ECDSA” is claimed in FCS_COP.1/SigGen, then “ECC–ERB” or “ECC–RS” must be claimed. The sizes of the private key, which is a scalar, and the public key, which is a point on the elliptic curve, are determined by the choice of the curve.

If the TSF acts only as a receiver in the RSA key establishment scheme, the ST does not need to implement RSA key generation.

Evaluation Activities

FCS_CKM.1/AKG

TSS

The evaluator shall examine the TSS to verify that it describes how the TOE generates a key based on output from a random bit generator as specified in FCS_RBG.1. The evaluator shall review the TSS to verify that it describes how the functionality described by FCS_RBG.1 is invoked.

The evaluator shall examine the TSS to verify that it identifies the usage and key lifecycle for keys generated using each selected algorithm.

The evaluator shall examine the TSS to verify that any one-time values such as nonces or masks are constructed in accordance with the relevant standards.

If the TOE uses the generated key in a key chain or hierarchy then the evaluator shall verify that the TSS describes how the key is used as part of the key chain or hierarchy.

Guidance

The evaluator shall verify that the guidance instructs the administrator how to configure the TOE to generate keys for the selected key generation algorithms for all key types and uses identified in the TSS.

Tests

The following tests are conditional based on the selections made in the SFR. The evaluator shall perform the following tests or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not

carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

RSA Key Generation

Identifier	***Cryptographic Key Generation Algorithm***	***Cryptographic Algorithm Parameters***	***List of Standards***
RSA	*RSA*	*Modulus of size [**selection:** 3072, 4096, 6144, 8192] bits*	*NIST FIPS PUB 186-5 (Section A.1.1)*

FIPS PUB 186-5 Key Pair generation specifies five methods for generating the primes pand q.

These are:

1. *Random provable primes*
2. *Random probable primes*
3. *Provable primes with conditions based on auxiliary provable primes*
4. *Probable primes with conditions based on auxiliary provable primes*
5. *Probable primes with conditions based on auxiliary probable primes*

In addition to the key generation method, the input parameters are:

- *Modulus [3072, 4096, 6144, 8192]*
- *Hash algorithm [SHA-384, SHA-512] (methods 1, 3, 4 only)*
- *Rabin-Miller prime test [2^{100}, $2^{Security\ String}$] (methods 2, 4, 5 only)*
- *p mod 8 value [0,1,3,5,7]*
- *q mod 8 value [0,1,3,5,7]*
- *Private key format [standard, Chinese Remainder Theorem]*
- *Public exponent [fixed value, random]*

The evaluator shall verify the ability of the TSF to correctly produce values for the RSA key components, including the public verification exponent e, the private prime factors pand q, the public modulus n, and the calculation of the private signature exponent d.

Testing for Random Provable Primes and Conditional Methods

To test the key generation method for the random provable primes method and for all the primes with conditions methods (methods 1, 3-5), the evaluator must seed the TSF key generation routine with sufficient data to deterministically generate the RSA key pair.

For each supported combination of the above input parameters, the evaluator shall have the TSF generate 25 key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated by a known good implementation using the same input parameters.

Testing for Random Probable Primes Method

If the TOE generates random probable primes (method 2) then, if possible, the random probable primes method should also be verified against a known good implementation as described above. If verification against a known good implementation is not possible, the evaluator shall have the TSF generate 25 key pairs for each supported key length nlen and verify that all of the following are true.

- *n = p*q*
- *p and q are probably prime according to Miller-Rabin tests with error probability <$2^{(-125)}$*
- *$2^{16} < e < 2^{256}$ and e is an odd integer*
- *GCD(p-1,e) = 1*
- *GCD(q-1,e) = 1*
- *$|p-q| > 2^{(nlen/2\ -\ 100)}$*
- *$p \geq squareroot(2)*(\ 2^{(nlen/2\ -1)}\ )$*
- *$q \geq squareroot(2)*(\ 2^{(nlen/2\ -1)}\ )$*
- *$2^{(nlen/2)} < d < LCM(p-1,q-1)$*
- *e*d = 1 mod LCM(p-1,q-1)*

Elliptic Curve Key Generation

Identifier	***Cryptographic Key Generation Algorithm***	***Cryptographic Algorithm Parameters***	***List of Standards***
ECC-ERB	*ECC – Extra Random Bits*	*Elliptic Curve [**selection:** P-384, P-521]*	*NIST FIPS PUB 186-5 (Section A.2.1)* *NIST SP 800-186 (Section 3) [NIST Curves]*
ECC-RS	*ECC – Rejection Sampling*	*Elliptic Curve [**selection:** P-384, P-521]*	*NIST FIPS PUB 186-5 (Section A.2.2)* *NIST SP 800-186 (Section 3) [NIST Curves]*

To test the TOE's ability to generate asymmetric cryptographic keys using elliptic curves, the evaluator shall perform the ECC Key Generation Test and the ECC Key Validation Test using the following input parameters.

- *Elliptic curve [P-384, P-521]*
- *Key pair generation method [extra random bits, rejection sampling]*

ECC Key Generation Test
For each supported combination of the above input parameters the evaluator shall require the implementation under test to generate 10 private and public key pairs (d, Q). The private key, d, shall be generated using a random bit generator as specified in FCS_RBG.1. The private key, d, is used to compute the public key, Q'. The evaluator shall confirm that 0< d< n(where nis the order of the group), and the computed value Q'is then compared to the generated public and private key pairs' public key, Q, to confirm that Qis equal to Q'.

ECC Key Validation Test
For each supported combination of the above parameters the evaluator shall generate 12 private and public key pairs using the key generation function of a known good implementation. For each set of 12 public keys, the evaluator shall modify four public key values by shifting xor yout of range by adding the order of the field and modify four other public key values by shifting xor yso that they are still in bounds, but not on the curve. The remaining public key values are left unchanged (i.e., correct). To determine correctness, the evaluator shall submit the public keys to the public key validation (PKV) function of the TOE and confirm that the results correspond as expected for the modified and unmodified values.

Finite Field Cryptography Key Generation

Identifier	***Cryptographic Key Generation Algorithm***	***Cryptographic Algorithm Parameters***	***List of Standards***
FFC-ERB	*FFC – Extra Random Bits*	*Static domain parameters approved for [**selection:** IKE groups [**selection:** MODP-3072, MODP-4096, MODP-6144, MODP-8192], TLS groups [**selection:** ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]]]*	*NIST SP 800-56A Revision 3 (Section 5.6.1.1.3) [key pair generation]* *[**selection:** RFC 3526 [IKE groups], RFC 7919 [TLS groups]]*
FFC-RS	*FFC – Rejection Sampling*	*Static domain parameters approved for [**selection:** IKE groups [**selection:** MODP-3072, MODP-4096, MODP-6144, MODP-8192], TLS groups [**selection:** ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]]]*	*NIST SP 800-56A Revision 3 (Section 5.6.1.1.4) [key pair generation]* *[**selection:** RFC 3526 [IKE groups], RFC 7919 [TLS groups]]*

To test the TOE's ability to generate asymmetric cryptographic keys using finite fields, the evaluator shall perform the Safe Primes Generation Test and the Safe Primes Validation Test using the following input parameter:

- *Fields/Groups [MODP-3072, MODP-4096, MODP-6144, MODP-8192, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]*

Safe Primes Generation Test
For each supported safe primes group, generate 10 key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated by a known good implementation using the same input parameters.

Safe Primes Verification Test
For each supported safe primes group, use a known good implementation to generate 10 key pairs. For each set of 10, the evaluator shall modify three so they are incorrect. The remaining values are left unmodified (i.e. correct). To determine correctness, the evaluator shall submit the key pairs to the public key validation (PKV) function of the TOE and shall confirm that the results correspond as expected for the modified and unmodified values.

LMS Key Generation

Identifier	***Cryptographic***	***Cryptographic Algorithm Parameters***	***List of***

	Key Generation Algorithm		Standards
LMS	*LMS Key Generation*	*Private key size = [**selection:** 192 bits with [**selection:** SHA-256/192, SHAKE256/192], 256 bits with [**selection:** SHA-256, SHAKE256]]; Winternitz parameter = [**selection:** 1, 2, 4, 8]; Tree height = [**selection:** 5, 10, 15, 20, 25]*	*RFC 8554 [LMS]* *NIST SP 800-208 [parameters]*

To test the TOE's ability to generate asymmetric cryptographic keys using LMS, the evaluator shall perform the LMS Key Generation Test using the following input parameters:

- *Hash algorithm [SHA-256/192, SHAKE256/192, SHA-256, SHAKE256]*
- *Winternitz [1, 2, 4, 8]*
- *Tree height [5, 10, 15, 20, 25]*

LMS Key Generation Test
For each supported combination of the hash algorithm, Winternitz parameter, and tree height, the evaluator shall generate one public key for each of the test cases. The number of test cases depends on the tree height:

Table 4: Number of LMS Test Cases

Height	Number of test cases
5	*5*
10	*4*
15	*3*
20	*2*
25	*1*

The evaluator shall verify the correctness of the TSF's implementation by comparing the public key generated by the TSF with that generated by a known good implementation using the same input parameters.

ML-KEM Key Generation

Identifier	Cryptographic Key Generation Algorithm	Cryptographic Algorithm Parameters	List of Standards
ML-KEM	*ML-KEM Key Generation*	*Parameter set = [ML-KEM-1024]*	*NIST FIPS PUB 203 (Section 7.1)*

To test the TOE's ability to generate asymmetric cryptographic keys using ML-KEM, the evaluator shall perform the Algorithm Functional Test using the following input parameters:

- *Parameter set [ML-KEM-1024]*
- *Random seed d [32 bytes]*
- *Random seed z [32 bytes]*

Algorithm Functional Test
For each supported parameter set the evaluator shall require the implementation under test to generate 25 key pairs using 25 different randomly generated pairs of 32-byte seed values (d, z). To determine correctness, the evaluator shall compare the resulting key pairs (ek, dk) with those generated using a known good implementation using the same inputs.

ML-DSA Key Generation

Identifier	Cryptographic Key Generation Algorithm	Cryptographic Algorithm Parameters	List of Standards
ML-DSA	*ML-DSA Key Generation*	*Parameter set = ML-DSA-87*	*NIST FIPS PUB 204 (Section 5.1)*

To test the TOE's ability to generate asymmetric cryptographic keys using ML-DSA, the evaluator shall perform the Algorithm Functional Test using the following input parameters:

- *Parameter set [ML-DSA-87]*
- *Random seed [32 bytes]*

Algorithm Functional Test
For each supported parameter set the evaluator shall require the implementation under test to generate 25 key pairs using 25 different randomly generated 32-byte seed values. To determine correctness, the evaluator shall compare the resulting key pairs with those generated using a known good implementation using the same inputs.

XMSS Key Generation

Identifier	*Cryptographic Key Generation Algorithm*	*Cryptographic Algorithm Parameters*	*List of Standards*
XMSS	*XMSS*	*Private key size = [**selection:** 192 bits with [**selection:** SHA-256/192, SHAKE256/192], 256 bits with [**selection:** SHA-256, SHAKE256]], tree height = [**selection:** 10, 16, 20]*	*RFC 8391 [XMSS]* *NIST SP 800-208 [parameters]*

To test the TOE's ability to generate asymmetric cryptographic keys using XMSS, the evaluator shall perform the XMSS Key Generation Test using the following input parameters:

- *Hash algorithm [SHA-256/192, SHAKE256/192, SHA-256, SHAKE256]*
- *Tree height [10, 16, 20] (XMSS only)*

Table 5: Number of Test Cases for XMSS^{MT}

Height	*Number of test cases*
10	*5*
16	*4*
20	*3*
40	*2*
60	*1*

XMSS Key Generation Test
For each supported combination of hash algorithm and tree height, the evaluator shall generate one public key for each test case. The number of test cases depends on the tree height as specified in Table 5.

The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated by a known good implementation using the same input parameters.

Note: The number of test cases is limited due to the extreme amount of time it can take to generate XMSS trees.

FCS_CKM.1/SKG Cryptographic Key Generation - Symmetric Key

FCS_CKM.1.1/SKG

The TSF shall generate **symmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm [**selection**: *Cryptographic Key Generation Algorithm*] and specified cryptographic key sizes [**selection**: *Cryptographic Key Sizes*] that meet the following: [**selection**: *List of standards*]

The following table provides the allowable choices for completion of the selection operations of FCS_CKM.1/SKG.

Table 6: Allowable Choices for FCS_CKM.1/SKG

Identifier	Cryptographic Key Generation Algorithm	Cryptographic Key Sizes	List of standards
RSK	Direct Generation from a Random Bit Generator as specified in FCS_RBG.1	[**selection**: *256*, *384*, *512*] bits	NIST SP 800-133 Revision 2 (Section 6.1)[Direct generation of symmetric keys]

Evaluation Activities

FCS_CKM.1/SKG
TSS
The evaluator shall examine the TSS to verify that it describes how the TOE obtains a symmetric cryptographic key through direct generation from a random bit generator as specified in FCS_RBG.1. The evaluator shall review the TSS to verify that it describes how the functionality described by FCS_RBG.1 is invoked.

The evaluator shall examine the TSS to verify that it identifies the usage, and key lifecycle for keys generated using each selected algorithm.

If the TOE uses the generated key in a key chain/hierarchy then the evaluator shall verify that the TSS describes how the key is used as part of the key chain/hierarchy.

Guidance
The evaluator shall verify that the AGD instructs the administrator how to configure the TOE to use the RBG to generate symmetric keys for all uses identified in the ST.

Tests
The following tests are conditional based upon the selections made in the SFR. The evaluator shall perform the following test or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

To test the TOE's ability to generate symmetric cryptographic keys using a random bit generator, the evaluator shall configure the symmetric cryptographic key generation capability for each claimed key size. The evaluator shall use the description of the RBG interface to verify that the TOE requests and receives an amount of RBG output greater than or equal to the requested key size.

FCS_CKM.2 Cryptographic Key Distribution

FCS_CKM.2.1

The TSF shall distribute cryptographic keys in accordance with the specified cryptographic key distribution method **key wrapping** and [**selection**: *key encapsulation*, ***no other methods***] that meets the following: [none].

Application Note: Key wrapping is used in support of wireless LAN communications. Key encapsulation is used in support of TLS when ML-KEM is used as the method of key establishment.

Evaluation Activities

FCS_CKM.2
TSS
The evaluator shall ensure that the TSS documents that the security strength supported by the selected key distribution methods is sufficient for the security strength of the keys distributed through those methods.

It is not necessary to identify the services that use each key distribution method here. That information should be documented in the requirements for the individual services and protocols that invoke key distribution.

Guidance
The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key distribution methods.

Tests
Specific testing for this component is covered by testing for the claimed components in FCS_COP.1/KeyEncap or FCS_COP.1/KeyWrap, depending on selections made.

FCS_CKM.6 Timing and Event of Cryptographic Key Destruction

FCS_CKM.6.1

The TSF shall destroy [*all plaintext keying material and critical security parameters (CSPs)*] when [**selection**: *no longer needed*, [***assignment**: other circumstances for destruction]*].

FCS_CKM.6.2

The TSF shall destroy **plaintext** keying material **and critical security parameters** by implementing key destruction in accordance with the following rules: [

- *For volatile memory, the destruction shall be executed by a single direct overwrite consisting of [**selection**: a pseudo-random pattern using a TSF DRBG (as specified in FCS_RBG.1), zeroes]*
- *For non-volatile EEPROM, the destruction shall be executed by a single direct overwrite consisting of a pseudo-random pattern using a TSF DRBG (as specified in FCS_RBG.1), followed by a read-verify.*
- *For non-volatile flash memory, that is not wear-leveled, the destruction shall be executed by [**selection**: a single direct overwrite consisting of zeros followed by a read-verify, a block erase that erases the reference to memory that stores data as well as the data itself]*
- *For non-volatile flash memory that is wear-leveled, the destruction shall be executed by [**selection**: a single direct overwrite consisting of zeros, a block erase]*
- *For non-volatile memory other than EEPROM and flash, the destruction*

shall be executed by a single direct overwrite with a random pattern that is changed before each write.

]

Application Note: For the purposes of this requirement, keying material refers to authentication data, passwords, secret/private symmetric keys, private asymmetric keys, data used to derive keys, values derived from passwords, etc. “Plaintext keying material” may refer to a KEK that is used to encrypt other keying material. Destruction of encrypted keying material may be accomplished by destroying the KEK used to encrypt it. If different mechanisms are used for destroying different keying material, all relevant claims should be selected and the TSS should identify which keying material is destroyed by which mechanism.

There are multiple situations in which plaintext keying material is no longer needed, including when the TOE is powered off, when the wipe when trusted channels are disconnected, when keying material is no longer needed by the trusted channel per the protocol, and when transitioning to the locked state (for those values derived from the Password Authentication Factor or that key material which is protected by the password-derived or biometric-unlocked KEK according to FCS_STG_EXT.2 – see Figure 3). For keys (or key material used to derive those keys) protecting sensitive data received in the locked state, "no longer needed" includes "while in the locked state." The assignment for "other circumstances for destruction" is completed for any user or Administrator directed key destruction such as initiating the wipe function or other manual destruction.

Key storage areas in non-volatile storage can be overwritten with any value that renders the keys unrecoverable. The value used can be all zeroes, all ones, or any other pattern or combination of values significantly different than the value of the key itself. When ‘a value that does not contain any CSP’ is chosen, it means that the TOE uses some other specified data not drawn from a source that may contain keying material or reveal information about it or any other TSF-protected data. In other words, the data used for overwriting is carefully selected and not taken from a general ‘pool’ that might contain current or residual data that itself requires confidentiality protection. If multiple copies exist, all copies must be destroyed.

Evaluation Activities

FCS_CKM.6

TSS

The evaluator shall verify that the TSS identifies all plaintext keying material and CSPs stored by the TOE, the type of memory in which it is stored, and when and how the keying material is erased. If the TOE uses one or more KEKs to protect stored keying material, the evaluator shall verify that the TSS describes the destruction of that keying material, either directly or by destruction of the KEK used to encrypt it.

If different types of memory are used to store the materials to be protected, the evaluator shall check to ensure that the TSS describes the clearing procedure in terms of the memory in which the data are stored (for example, "secret keys stored on flash are cleared by overwriting once with zeros, while secret keys stored on the internal persistent storage device are cleared by overwriting one time with a random pattern that is changed before each write"). For block erases, the evaluator shall also ensure that the TSS identifies the block erase command that is used and shall verify that the command used also addresses any copies of the plaintext key material that may be created, e.g. in order to optimize the use of Flash memory.

Guidance

There are no additional Guidance evaluation activities for this component.

Tests

Evaluation Activity Note: *The following tests likely require the TOE developer to provide access to a test platform that provides the evaluator with tools that are typically not found on the end consumer version of the TOE.*

The evaluator shall perform the following tests for all keys and key material subject to destruction by the TOE. For these tests, the evaluator shall utilize an appropriate development environment (e.g., a virtual machine) and development tools (debuggers, simulators, etc.) to test that keys are cleared, including all copies of the key that may have been created internally by the TOE during normal cryptographic processing with that key.

- *Test FCS_CKM.6:1: Applied to each key held as plaintext in volatile memory and subject to destruction by overwrite by the TOE (whether or not the plaintext value is subsequently encrypted for storage in volatile or non-volatile memory). In the case where the only selection made for the destruction method key was removal of power, then this test is unnecessary. The evaluator shall:*
  1. *Record the value of the key in the TOE subject to clearing.*
  2. *Cause the TOE to perform a normal cryptographic processing with the key from Step #1.*
  3. *Cause the TOE to clear the key.*
  4. *Cause the TOE to stop the execution but not exit.*
  5. *Cause the TOE to dump the entire memory of the TOE into a binary file.*
  6. *Search the content of the binary file created in Step #5 for instances of the known key*

value from Step #1.
7. *Break the key value from Step #1 into 3 similar sized pieces and perform a search using each piece.*

Steps 1-6 ensure that the complete key does not exist anywhere in volatile memory. If a copy is found, then the test fails.

Step 7 ensures that partial key fragments do not remain in memory. If a fragment is found, there is a minuscule chance that it is not within the context of a key (e.g., some random bits that happen to match). If this is the case the test should be repeated with a different key in Step #1. If a fragment is found the test fails.

- *Test FCS_CKM.6:2: Applied to each key held in non-volatile memory and subject to destruction by overwrite by the TOE. The evaluator shall use special tools (as needed), provided by the TOE developer if necessary, to view the key storage location:*
  1. *Record the value of the key in the TOE subject to clearing.*
  2. *Cause the TOE to perform a normal cryptographic processing with the key from Step #1.*
  3. *Cause the TOE to clear the key.*
  4. *Search the non-volatile memory the key was stored in for instances of the known key value from Step #1. If a copy is found, then the test fails.*
  5. *Break the key value from Step #1 into 3 similar sized pieces and perform a search using each piece. If a fragment is found then the test is repeated (as described for test 1 above), and if a fragment is found in the repeated test then the test fails.*

- *Test FCS_CKM.6:3: Applied to each key held as non-volatile memory and subject to destruction by overwrite by the TOE. The evaluator shall use special tools (as needed), provided by the TOE developer if necessary, to view the key storage location:*
  1. *Record the storage location of the key in the TOE subject to clearing.*
  2. *Cause the TOE to perform a normal cryptographic processing with the key from Step #1.*
  3. *Cause the TOE to clear the key.*
  4. *Read the storage location in Step #1 of non-volatile memory to ensure the appropriate pattern is utilized.*

The test succeeds if correct pattern is used to overwrite the key in the memory location. If the pattern is not found the test fails.

FCS_CKM_EXT.1 Cryptographic Key Support

FCS_CKM_EXT.1.1

The TSF shall support [**selection**: *immutable hardware*, *mutable hardware*] REKs with a [**selection**: *symmetric*, *asymmetric*] key of strength [**selection**: *192 bits*, *256 bits*].

FCS_CKM_EXT.1.2

Each REK shall be hardware-isolated from the OS on the TSF in runtime.

FCS_CKM_EXT.1.3

Each REK shall be generated by an RBG in accordance with FCS_RBG.1.

Application Note: Either asymmetric or symmetric keys are allowed; the ST author makes the selection appropriate for the device. Symmetric keys must be of size 256 bits in order to correspond with FCS_COP.1/SKC. Asymmetric keys may be of any strength corresponding to FCS_CKM.1/AKG.

The raw key material of "immutable hardware" REKs is computationally processed by hardware and software cannot access the raw key material. Thus if immutable hardware is selected in FCS_CKM_EXT.1.1 it implicitly meets FCS_CKM_EXT.7/LOCKED or FCS_CKM_EXT.7/UNLOCKED. If mutable hardware is selected in FCS_CKM_EXT.1.1, FCS_CKM_EXT.7/LOCKED or FCS_CKM_EXT.7/UNLOCKED must be included in the ST.

The lack of a public/documented API for importing or exporting the REK, when a private/undocumented API exists, is not sufficient to meet this requirement.

The RBG used to generate a REK may be an RBG native to the hardware key container or may be an off-device RBG. If performed by an off-device RBG, the device manufacturer must not be able to access a REK after the manufacturing process has been completed. The Evaluation Activities for these two cases differ.

Evaluation Activities

FCS_CKM_EXT.1
TSS
The evaluator shall review the TSS to determine that a REK is supported by the TOE, that the TSS includes a description of the protection provided by the TOE for a REK, and that the TSS

includes a description of the method of generation of a REK.

The evaluator shall verify that the description of the protection of a REK describes how any reading, import, and export of that REK is prevented. For example, if the hardware protecting the REK is removable, the description should include how other devices are prevented from reading the REK. The evaluator shall verify that the TSS describes how encryption/decryption/derivation actions are isolated so as to prevent applications and system-level processes from reading the REK while allowing encryption/decryption/derivation by the key.

The evaluator shall verify that the description includes how the OS is prevented from accessing the memory containing REK key material, which software is allowed access to the REK, how any other software in the execution environment is prevented from reading that key material, and what other mechanisms prevent the REK key material from being written to shared memory locations between the OS and the separate execution environment.

If key derivation is performed using a REK, the evaluator shall ensure that the TSS description includes a description of the key derivation function and shall verify the key derivation uses an approved derivation mode and key expansion algorithm according to FCS_CKM_EXT.3.2.

The evaluator shall verify that the generation of a REK meets the FCS_RBG.1 requirement:

- *If REKs is/are generated on-device, the TSS shall include a description of the generation mechanism including what triggers a generation, how the functionality described by FCS_RBG.1 is invoked, and whether a separate instance of the RBG is used for REKs.*
- *If REKs is/are generated off-device, the TSS shall include evidence that the RBG meets FCS_RBG.1. This will likely necessitate a second set of RBG documentation equivalent to the documentation provided for the RBG Evaluation Activities. In addition, the TSS shall describe the manufacturing process that prevents the device manufacturer from accessing any REKs.*

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FCS_CKM_EXT.2 Cryptographic Key Random Generation

FCS_CKM_EXT.2.1

All DEKs shall be [**selection**:

- *randomly generated*
- *from the combination of a randomly generated DEK with another DEK or salt in a way that preserves the effective entropy of each factor by [**selection**: using an XOR operation, concatenating the keys and using a KDF (as described in SP 800-108), concatenating the keys and using a KDF (as described in SP 800-56C)]*

] with entropy corresponding to the security strength of AES key sizes of 256 bits.

Application Note: The intent of this requirement is to ensure that the DEK cannot be recovered with less work than a full exhaust of the key space for AES. The key generation capability of the TOE uses an RBG implemented on the TOE device (FCS_RBG.1). A DEK is used in addition to the KEK so that authentication factors can be changed without having to re-encrypt all of the user data on the device.

The ST author selects all applicable DEK generation types implemented by the TOE.

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

Evaluation Activities

FCS_CKM_EXT.2
TSS
The evaluator shall ensure that the documentation of the product's encryption key management is detailed enough that, after reading, the product's key management hierarchy is clear and that it meets the requirements to ensure the keys are adequately protected. The evaluator shall ensure that the documentation includes both an essay and one or more diagrams. Note that this may also be documented as separate proprietary evidence rather than being included in the TSS.

The evaluator shall also examine the key hierarchy section of the TSS to ensure that the formation of all DEKs is described and that the key sizes match that described by the ST author. The evaluator shall examine the key hierarchy section of the TSS to ensure that each DEK is generated or combined from keys of equal or greater security strength using one of the selected methods.

- *If the symmetric DEK is generated by an RBG, the evaluator shall review the TSS to determine that it describes how the functionality described by FCS_RBG.1 is invoked. The evaluator uses the description of the RBG functionality in FCS_RBG.1 or documentation available for the operational environment to determine that the key size being requested is greater than or equal to the key size and mode to be used for the encryption/decryption of the data.*
- *If the DEK is formed from a combination, the evaluator shall verify that the TSS describes the method of combination and that this method is either an XOR or a KDF to justify that the effective entropy of each factor is preserved. The evaluator shall also verify that each combined value was originally generated from an Approved DRBG described in FCS_RBG.1.*
- *If concatenating the keys and using a KDF (as described in SP 800-56C) is selected, the evaluator shall ensure the TSS includes a description of the randomness extraction step.*

The description must include how an approved untruncated MAC function is being used for the randomness extraction step and the evaluator must verify the TSS describes that the output length (in bits) of the MAC function is at least as large as the targeted security strength (in bits) of the parameter set employed by the key establishment scheme (see Tables 1-3 of SP 800-56C).

The description must include how the MAC function being used for the randomness extraction step is related to the PRF used in the key expansion and verify the TSS description includes the correct MAC function:

- *If an HMAC-hash is used in the randomness extraction step, then the same HMAC-hash (with the same hash function hash) is used as the PRF in the key expansion step.*
- *If an AES-CMAC is used in the randomness extraction step, then AES-CMAC with a 128-bit key is used as the PRF in the key expansion step.*
- *The description must include the lengths of the salt values being used in the randomness extraction step and the evaluator shall verify the TSS description includes correct salt lengths:*
- *If an HMAC-hash is being used as the MAC, the salt length can be any value up to the maximum bit length permitted for input to the hash function hash.*
- *If an AES-CMAC is being used as the MAC, the salt length shall be the same length as the AES key (i.e., 256 bits).*

(conditional) If a KDF is used, the evaluator shall ensure that the TSS includes a description of the key derivation function and shall verify the key derivation uses an approved derivation mode and key expansion algorithm according to SP 800-108 or SP 800-56C.

Guidance
The evaluator uses the description of the RBG functionality in FCS_RBG.1 or documentation available for the operational environment to determine that the key size being generated or combined is identical to the key size and mode to be used for the encryption/decryption of the data.

Tests
If a KDF is used, the evaluator shall perform one or more of the following tests to verify the correctness of the key derivation function, depending on the modes that are supported. Table 7 maps the data fields to the notations used in SP 800-108 and SP 800-56C.

Table 7: Notations used in SP 800-108 and SP 800-56C

Data Fields	***Notations***	
	SP 800-108	***SP 800-56C***
Pseudorandom function	*PRF*	*PRF*
Counter length	*r*	*r*
Length of output of PRF	*h*	*h*
Length of derived keying material	*L*	*L*
Length of input values	*l length*	*l length*
Pseudorandom input values I	*K1 (key derivation key)*	*Z (shared secret)*
Pseudorandom salt values	*n/a*	*s*
Randomness extraction MAC	*n/a*	*MAC*

Counter Mode Tests:

The evaluator shall determine the following characteristics of the key derivation function:

- *One or more pseudorandom functions that are supported by the implementation (PRF).*
- *One or more of the values {8, 16, 24, 32} that equal the length of the binary representation of the counter (r).*
- *The length (in bits) of the output of the PRF (h).*
- *Minimum and maximum values for the length (in bits) of the derived keying material (L). These values can be equal if only one value of L is supported. These must be evenly divisible by h.*
- *Up to two values of L that are NOT evenly divisible by h.*
- *Location of the counter relative to fixed input data: before, after, or in the middle.*
    - *Counter before fixed input data: fixed input data string length (in bytes), fixed input data string value.*
    - *Counter after fixed input data: fixed input data string length (in bytes), fixed input data string value.*
    - *Counter in the middle of fixed input data: length of data before counter (in bytes), length of data after counter (in bytes), value of string input before counter, value of string input after counter.*
- *The length (I_length) of the input values I.*

For each supported combination of I_length, MAC, salt, PRF, counter location, value of r, and value of L, the evaluator shall generate 10 test vectors that include pseudorandom input values I, and pseudorandom salt values. If there is only one value of L that is evenly divisible by h, the evaluator shall generate 20 test vectors for it. For each test vector, the evaluator shall supply this data to the TOE in order to produce the keying material output.

The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

Feedback Mode Tests:

The evaluator shall determine the following characteristics of the key derivation function:

- *One or more pseudorandom functions that are supported by the implementation (PRF).*
- *The length (in bits) of the output of the PRF (h).*
- *Minimum and maximum values for the length (in bits) of the derived keying material (L). These values can be equal if only one value of L is supported. These must be evenly divisible by h.*
- *Up to two values of L that are NOT evenly divisible by h.*
- *Whether or not zero-length IVs are supported.*
- *Whether or not a counter is used, and if so:*
    - *One or more of the values {8, 16, 24, 32} that equal the length of the binary representation of the counter (r).*
    - *Location of the counter relative to fixed input data: before, after, or in the middle.*
        - *Counter before fixed input data: fixed input data string length (in bytes), fixed input data string value.*
        - *Counter after fixed input data: fixed input data string length (in bytes), fixed input data string value.*
        - *Counter in the middle of fixed input data: length of data before counter (in bytes), length of data after counter (in bytes), value of string input before counter, value of string input after counter.*
- *The length (I_length) of the input values I.*

For each supported combination of I_length, MAC, salt, PRF, counter location (if a counter is used), value of r (if a counter is used), and value of L, the evaluator shall generate 10 test vectors that include pseudorandom input values I and pseudorandom salt values. If the KDF supports zero-length IVs, five of these test vectors will be accompanied by pseudorandom IVs and the other five will use zero-length IVs. If zero-length IVs are not supported, each test vector will be accompanied by an pseudorandom IV. If there is only one value of L that is evenly divisible by h, the evaluator shall generate 20 test vectors for it.

For each test vector, the evaluator shall supply this data to the TOE in order to produce the keying material output. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

Double Pipeline Iteration Mode Tests:

The evaluator shall determine the following characteristics of the key derivation function:

- *One or more pseudorandom functions that are supported by the implementation (PRF).*
- *The length (in bits) of the output of the PRF (h).*
- *Minimum and maximum values for the length (in bits) of the derived keying material (L). These values can be equal if only one value of L is supported. These must be evenly divisible by h.*
- *Up to two values of L that are NOT evenly divisible by h.*
- *Whether or not a counter is used, and if so:*
    - *One or more of the values {8, 16, 24, 32} that equal the length of the binary representation of the counter (r).*
    - *Location of the counter relative to fixed input data: before, after, or in the middle.*
        - *Counter before fixed input data: fixed input data string length (in bytes), fixed*

input data string value.
  - *Counter after fixed input data: fixed input data string length (in bytes), fixed input data string value.*
  - *Counter in the middle of fixed input data: length of data before counter (in bytes), length of data after counter (in bytes), value of string input before counter, value of string input after counter.*
- *The length (I_length) of the input values I.*

For each supported combination of I_length, MAC, salt, PRF, counter location (if a counter is used), value of r (if a counter is used), and value of L, the evaluator shall generate 10 test vectors that include pseudorandom input values I, and pseudorandom salt values. If there is only one value of L that is evenly divisible by h, the evaluator shall generate 20 test vectors for it.

For each test vector, the evaluator shall supply this data to the TOE in order to produce the keying material output. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

FCS_CKM_EXT.3 Cryptographic Key Generation

FCS_CKM_EXT.3.1

The TSF shall use [**selection**: *asymmetric KEKs of [**assignment**: security strength greater than or equal to 192 bits] security strength, symmetric KEKs of 256-bit security strength corresponding to at least the security strength of the keys encrypted by the KEK*].

Application Note: The ST author selects all applicable KEK types implemented by the TOE.

FCS_CKM_EXT.3.2

The TSF shall generate all KEKs using one of the following methods:

- Derive the KEK from a Password Authentication Factor according to FCS_CKM_EXT.8 and

[**selection**:

- *Generate the KEK using an RBG that meets this profile (as specified in FCS_RBG.1)*
- *Generate the KEK using a key generation scheme that meets this profile (as specified in FCS_CKM.1/AKG or FCS_CKM.1/SKG)*
- *Combine the KEK from other KEKs in a way that preserves the effective entropy of each factor by [**selection**: using an XOR operation, concatenating the keys and using a KDF (as described in SP 800-108), concatenating the keys and using a KDF (as described in SP 800-56C), encrypting one key with another]*

].

Application Note: The conditioning of passwords is performed in accordance with FCS_CKM_EXT.8.

It is expected that key generation derived from conditioning, using an RBG or generation scheme, and through combination, will each be necessary to meet the requirements set out in this document. In particular, Figure 3 has KEKs of each type: KEK_3 is generated, KEK_1 is derived from a Password Authentication Factor, and KEK_2 is combined from two KEKs. In Figure 3, KEK_3 may either be a symmetric key generated from an RBG or an asymmetric key generated using a key generation scheme according to FCS_CKM.1/AKG.

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

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

Evaluation Activities

FCS_CKM_EXT.3
TSS
The evaluator shall examine the key hierarchy section of the TSS to ensure that the formation of all KEKs are described and that the key sizes match that described by the ST author. The evaluator shall examine the key hierarchy section of the TSS to ensure that each key (DEKs, software-based key storage, and KEKs) is encrypted by keys of equal or greater security strength using one of the selected methods.

The evaluator shall review the TSS to verify that it contains a description of the conditioning used to derive KEKs. This description must include the size and storage location of salts. This activity may be performed in combination with that for FCS_CKM_EXT.8.

(conditional) If the symmetric KEK is generated by an RBG, the evaluator shall review the TSS to determine that it describes how the functionality described by FCS_RBG.1 is invoked. The evaluator uses the description of the RBG functionality in FCS_RBG.1 or documentation available for the operational environment to determine that the key size being requested is greater than or equal to the key size and mode to be used for the encryption/decryption of the data.

(conditional) If the KEK is generated according to an asymmetric key scheme, the evaluator shall review the TSS to determine that it describes how the functionality described by FCS_CKM.1/AKG is invoked. The evaluator uses the description of the key generation functionality in FCS_CKM.1/AKG or documentation available for the operational environment to determine that the key strength being requested is greater than or equal to 112 bits.

(conditional) If the KEK is formed from a combination, the evaluator shall verify that the TSS describes the method of combination and that this method is either an XOR, a KDF, or encryption.

(conditional) If a KDF is used, the evaluator shall ensure that the TSS includes a description of the key derivation function and shall verify the key derivation uses an approved derivation mode and key expansion algorithm according to SP 800-108.

(conditional) If concatenating the keys and using a KDF (as described in SP 800-56C) is selected, the evaluator shall ensure the TSS includes a description of the randomness extraction step. The description must include

- *How an approved untruncated MAC function is being used for the randomness extraction step and the evaluator must verify the TSS describes that the output length (in bits) of the MAC function is at least as large as the targeted security strength (in bits) of the parameter set employed by the key establishment scheme (see Tables 1-3 of SP 800-56C).*
- *How the MAC function being used for the randomness extraction step is related to the PRF used in the key expansion and verify the TSS description includes the correct MAC function:*
  - *If an HMAC-hash is used in the randomness extraction step, then the same HMAC-hash (with the same hash function hash) is used as the PRF in the key expansion step.*
  - *If an AES-CMAC (with key length 128, 192, or 256 bits) is used in the randomness extraction step, then AES-CMAC with a 128-bit key is used as the PRF in the key expansion step.*
- *The lengths of the salt values being used in the randomness extraction step and the evaluator shall verify the TSS description includes correct salt lengths:*
  - *If an HMAC-hash is being used as the MAC, the salt length can be any value up to the maximum bit length permitted for input to the hash function hash.*
  - *If an AES-CMAC is being used as the MAC, the salt length shall be the same length as the AES key (i.e., 256 bits).*

The evaluator shall also ensure that the documentation of the product's encryption key management is detailed enough that, after reading, the product's key management hierarchy is clear and that it meets the requirements to ensure the keys are adequately protected. The evaluator shall ensure that the documentation includes both an essay and one or more diagrams. Note that this may also be documented as separate proprietary evidence rather than being included in the TSS.

Guidance
There are no guidance evaluation activities for this component.

Tests
If a KDF is used, the evaluator shall perform one or more of the following tests to verify the correctness of the key derivation function, depending on the modes that are supported. Table 8 maps the data fields to the notations used in SP 800-108 and SP 800-56C.

Table 8: Notations used in SP 800-108 and SP 800-56C

Data Fields	***Notations***	
	SP 800-108	***SP 800-56C***
Pseudorandom function	*PRF*	*PRF*
Counter length	*r*	*r*
Length of output of PRF	*h*	*h*
Length of derived keying material	*L*	*L*
Length of input values	*I_length*	*I_length*

Pseudorandom input values I	*K_1 (key derivation key)*	*Z (shared secret)*
Pseudorandom salt values	*n/a*	*s*
Randomness extraction MAC	*n/a*	*MAC*

Counter Mode Tests:

The evaluator shall determine the following characteristics of the key derivation function:

- *One or more pseudorandom functions that are supported by the implementation (PRF).*
- *One or more of the values {8, 16, 24, 32} that equal the length of the binary representation of the counter (r).*
- *The length (in bits) of the output of the PRF (h).*
- *Minimum and maximum values for the length (in bits) of the derived keying material (L). These values can be equal if only one value of L is supported. These must be evenly divisible by h.*
- *Up to two values of L that are NOT evenly divisible by h.*
- *Location of the counter relative to fixed input data: before, after, or in the middle.*
  - *Counter before fixed input data: fixed input data string length (in bytes), fixed input data string value.*
  - *Counter after fixed input data: fixed input data string length (in bytes), fixed input data string value.*
  - *Counter in the middle of fixed input data: length of data before counter (in bytes), length of data after counter (in bytes), value of string input before counter, value of string input after counter.*
- *The length (I_length) of the input values I.*

For each supported combination of I_length, MAC, salt, PRF, counter location, value of r, and value of L, the evaluator shall generate 10 test vectors that include pseudorandom input values I, and pseudorandom salt values. If there is only one value of L that is evenly divisible by h, the evaluator shall generate 20 test vectors for it. For each test vector, the evaluator shall supply this data to the TOE in order to produce the keying material output.

The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

Feedback Mode Tests:
The evaluator shall determine the following characteristics of the key derivation function:

- *One or more pseudorandom functions that are supported by the implementation (PRF).*
- *The length (in bits) of the output of the PRF (h).*
- *Minimum and maximum values for the length (in bits) of the derived keying material (L). These values can be equal if only one value of L is supported. These must be evenly divisible by h.*
- *Up to two values of L that are NOT evenly divisible by h.*
- *Whether or not zero-length IVs are supported.*
- *Whether or not a counter is used, and if so:*
  - *One or more of the values {8, 16, 24, 32} that equal the length of the binary representation of the counter (r).*
  - *Location of the counter relative to fixed input data: before, after, or in the middle.*
    - *Counter before fixed input data: fixed input data string length (in bytes), fixed input data string value.*
    - *Counter after fixed input data: fixed input data string length (in bytes), fixed input data string value.*
    - *Counter in the middle of fixed input data: length of data before counter (in bytes), length of data after counter (in bytes), value of string input before counter, value of string input after counter.*
- *The length (I_length) of the input values I.*

For each supported combination of I_length, MAC, salt, PRF, counter location (if a counter is used), value of r (if a counter is used), and value of L, the evaluator shall generate 10 test vectors that include pseudorandom input values I and pseudorandom salt values. If the KDF supports zero-length IVs, five of these test vectors will be accompanied by pseudorandom IVs and the other five will use zero-length IVs. If zero-length IVs are not supported, each test vector will be accompanied by an pseudorandom IV. If there is only one value of L that is evenly divisible by h, the evaluator shall generate 20 test vectors for it.

For each test vector, the evaluator shall supply this data to the TOE in order to produce the keying material output. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

Double Pipeline Iteration Mode Tests:
The evaluator shall determine the following characteristics of the key derivation function:

- *One or more pseudorandom functions that are supported by the implementation (PRF).*

- *The length (in bits) of the output of the PRF (h).*
- *Minimum and maximum values for the length (in bits) of the derived keying material (L). These values can be equal if only one value of L is supported. These must be evenly divisible by h.*
- *Up to two values of L that are NOT evenly divisible by h.*
- *Whether or not a counter is used, and if so:*
  - *One or more of the values {8, 16, 24, 32} that equal the length of the binary representation of the counter (r).*
  - *Location of the counter relative to fixed input data: before, after, or in the middle.*
    - *Counter before fixed input data: fixed input data string length (in bytes), fixed input data string value.*
    - *Counter after fixed input data: fixed input data string length (in bytes), fixed input data string value.*
    - *Counter in the middle of fixed input data: length of data before counter (in bytes), length of data after counter (in bytes), value of string input before counter, value of string input after counter.*
- *The length (I_length) of the input values I.*

For each supported combination of I_length, MAC, salt, PRF, counter location (if a counter is used), value of r (if a counter is used), and value of L, the evaluator shall generate 10 test vectors that include pseudorandom input values I, and pseudorandom salt values. If there is only one value of L that is evenly divisible by h, the evaluator shall generate 20 test vectors for it.

For each test vector, the evaluator shall supply this data to the TOE in order to produce the keying material output. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

FCS_CKM_EXT.5 TSF Wipe

FCS_CKM_EXT.5.1

The TSF shall wipe all protected data by [**selection**:

- *Cryptographically erasing the encrypted DEKs or the KEKs in non-volatile memory by following the requirements in FCS_CKM_EXT.6.2*
- *Overwriting all PD according to the following rules:*
  - *For EEPROM, the destruction shall be executed by a single direct overwrite consisting of a pseudo random pattern using the TSF's RBG (as specified in FCS_RBG.1, followed by a read-verify.*
  - *For flash memory, that is not wear-leveled, the destruction shall be executed [**selection**: by a single direct overwrite consisting of zeros followed by a read-verify, by a block erase that erases the reference to memory that stores data as well as the data itself].*
  - *For flash memory, that is wear-leveled, the destruction shall be executed [**selection**: by a single direct overwrite consisting of zeros, by a block erase].*
  - *For non-volatile memory other than EEPROM and flash, the destruction shall be executed by a single direct overwrite with a random pattern that is changed before each write.*

].

Application Note: Protected data is all non-TSF data, including all user or enterprise data. Some or all of this data may be considered sensitive data as well.

FCS_CKM_EXT.5.2

The TSF shall perform a power cycle on conclusion of the wipe procedure.

Evaluation Activities

FCS_CKM_EXT.5
TSS
The evaluator shall check to ensure the TSS describes how the device is wiped, the type of clearing procedure that is performed (cryptographic erase or overwrite) and, if overwrite is performed, the overwrite procedure (overwrite with zeros, overwrite three or more times by a different alternating pattern, overwrite with random pattern, or block erase).

If different types of memory are used to store the data to be protected, the evaluator shall check to ensure that the TSS describes the clearing procedure in terms of the memory in which the data are stored (for example, data stored on flash are cleared by overwriting once with zeros, while data stored on the internal persistent storage device are cleared by overwriting three times with a random pattern that is changed before each write).

Guidance
The evaluator shall verify that the AGD guidance describes how to enable encryption, if it is not

enabled by default. Additionally the evaluator shall verify that the AGD guidance describes how to initiate the wipe command.

Tests
Evaluation Activity Note: *The following test may require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile Device products.*

- *Test FCS_CKM_EXT.5:1: The evaluator shall perform one of the following tests. The test before and after the wipe command shall be identical. This test shall be repeated for each type of memory used to store the data to be protected.*
    - *Test FCS_CKM_EXT.5:1.1:* ***For File-based Methods:***
      The evaluator shall enable encryption according to the AGD guidance. The evaluator shall create a user data (protected data or sensitive data) file, for example, by using an application. The evaluator shall use a tool provided by the developer to examine this data stored in memory (for example, by examining a decrypted files). The evaluator shall initiate the wipe command according to the AGD guidance provided for FMT_SMF_EXT.1. The evaluator shall use a tool provided by the developer to examine the same data location in memory to verify that the data has been wiped according to the method described in the TSS (for example, the files are still encrypted and cannot be accessed).
    - *Test FCS_CKM_EXT.5:1.2:* ***For Volume-based Methods:***
      The evaluator shall enable encryption according to the AGD guidance. The evaluator shall create a unique data string, for example, by using an application. The evaluator shall use a tool provided by the developer to search decrypted data for the unique string. The evaluator shall initiate the wipe command according to the AGD guidance provided for FMT_SMF_EXT.1. The evaluator shall use a tool provided by the developer to search for the same unique string in decrypted memory to verify that the data has been wiped according to the method described in the TSS (for example, the files are still encrypted and cannot be accessed).
- *Test FCS_CKM_EXT.5:2: The evaluator shall cause the device to wipe and verify that the wipe concludes with a power cycle.*

FCS_CKM_EXT.6 Salt Generation

FCS_CKM_EXT.6.1

The TSF shall generate all salts using an RBG that meets FCS_RBG.1.

Application Note: This requirement refers only to salt generation. In the examples given, a salt may be used as part of the scheme/algorithm. Requirements on nonces or ephemeral keys are provided elsewhere, if needed. The list below is provided for clarity, in order to give examples of where the TSF may be generating cryptographic salts; it is not exhaustive nor is it intended to mandate implementation of all of these schemes/algorithms. Cryptographic salts are generated for various uses including:

- RSASSA-PSS signature generation
- DSA signature generation
- ECDSA signature generation
- DH static key agreement scheme
- PBKDF
- Key Agreement Scheme in NIST SP 800-56B
- AES GCM

Evaluation Activities

FCS_CKM_EXT.6
TSS
The evaluator shall verify that the TSS contains a description regarding the salt generation, including which algorithms on the TOE require salts. The evaluator shall confirm that the salt is generated using an RBG described in FCS_RBG.1. For PBKDF derivation of KEKs, this evaluation activity may be performed in conjunction with FCS_CKM_EXT.3.2.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FCS_CKM_EXT.7/LOCKED Cryptographic Key Establishment (When Locked)

FCS_CKM_EXT.7.1/LOCKED

The TSF shall derive shared cryptographic keys with input from multiple parties in accordance with specified cryptographic key agreement algorithms **for the purposes of encrypting sensitive data received while the device is locked**. [**selection**: *Cryptographic Algorithm*] and specified cryptographic parameters [**selection**: *Cryptographic Parameters*] that meet the following: [**selection**: *List of Standards*] .

The following table provides the allowable choices for completion of the selection operations of FCS_CKM_EXT.7/LOCKED.

Table 9: Allowable choices for FCS_CKM_EXT.7

Identifier	Cryptographic Algorithm	Cryptographic Parameters	List of Standards
KAS2	RSA	Modulus size [**selection**: *3072, 4096, 6144, 8192*] bits	NIST SP 800-56B Revision 2 (Section 8.3) [KAS2]
DH	Finite Field Cryptography Diffie-Hellman	Static domain parameters approved for [**selection**: • *IKE Groups [**selection**: MODP-3072, MODP-4096, MODP-6144, MODP-8192]* • *TLS Groups [**selection**: ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]*]	NIST SP 800-56A Revision 3 (Section 5.7.1.1) [DH] [**selection**: *RFC 3526 [IKE groups], RFC 7919 [TLS groups]*]
ECDH	Elliptic Curve Diffie-Hellman	Elliptic Curve [**selection**: *P-384, P-521*]	NIST SP 800-56A Revision 3 (Section 5.7.1.2) [ECDH] NIST SP 800-186 (Section 3.2.1) [NIST Curves]
ECDH-Ed	ECDH with Montgomery Curves	Domain parameters approved for elliptic curves []	RFC 7748 (Section 5) [ECDH-Ed] NIST SP 800-186 (Section 3.2.2) [Montgomery Curves]

Application Note: This SFR defines the asymmetric key scheme used to protect data at rest as defined by FDP_DAR_EXT.2.2.

Evaluation Activities

FCS_CKM_EXT.7/LOCKED
TSS
The evaluator shall ensure that the TSS documents that the security strength of the material contributed by the TOE is sufficient for the security strength of the key and the agreement method.

Guidance
There are no additional guidance evaluation activities for this component.

Tests
The following tests are conditional based upon the selections made in the SFR. The evaluator shall perform the following test or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

KAS2

Identifier	*Cryptographic Algorithm*	*Cryptographic Parameters*	*List of Standards*
KAS2	*RSA*	*Modulus Size [**selection:** 3072, 4096, 6144, 8192] bits*	*NIST SP 800-56B Revision 2 (Section 8.3) [KAS2]*

To test the TOE's implementation of the of the KAS2 RSA Key Agreement scheme, the evaluator

shall perform the Algorithm Functional Test and Validation Test using the following input parameters:

- *RSA Private key format [Basic, Prime Factor, Chinese Remainder Theorem]*
- *Modulo value [3072, 4096, 6144, 8192]*
- *Role [initiator, responder]*

The evaluator shall generate a test group (i.e. set of tests) for each parameter value of the above parameter type with the largest number of supported values. For example, if the TOE supports all five Modulo values, then the evaluator shall generate five test groups. Each of the above supported parameter values must be included in at least one test group.

Regardless of how many parameter values are supported, there must be at least two test groups.

Half of the test groups are designated as Algorithm Functional Tests (AFT) and the remainder are designated as Validation Tests (VAT). If there is an odd number of groups, then the extra group is designated randomly as either AFT or VAT.

Algorithm Functional Test
For each test group designated as AFT, the evaluator shall generate 10 test cases using random data (except for a fixed public exponent, if supported). The resulting shared secrets shall be compared with those generated by a known-good implementation using the same inputs.

Validation Test
For each test group designated as VAT, the evaluator shall generate 25 test cases are using random data (except for a fixed public exponent, if supported). Of the 25 test cases:

- *Two test cases must have a shared secret with a leading nibble of 0s,*
- *Two test cases have modified derived key material,*
- *Two test cases have modified tags, if key confirmation is supported,*
- *Two test cases have modified MACs, if key confirmation is supported, and*
- *The remaining test cases are not modified.*

To determine correctness, the evaluator shall confirm that the resulting 25 shared secrets correspond as expected for both the modified and unmodified values.

FFC Diffie-Hellman Key Agreement

Identifier	***Cryptographic Algorithm***	***Cryptographic Parameters***	***List of Standards***
DH	*Finite Field Cryptography Diffie-Hellman*	*Static domain parameters approved for [**selection:** IKE groups [**selection:** MODP-3072, MODP-4096, MODP-6144, MODP-8192], TLS groups [**selection:** ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]]]*	*NIST SP 800-56A Revision 3 (Section 5.7.1.1) [DH]* *[**selection:** RFC 3526 [IKE Groups], RFC 7919 [TLS Groups]]*

To test the TOE's implementation of FFC Diffie-Hellman Key Agreement, the evaluator shall perform the Algorithm Functional Test and Validation Test using the following input parameters:

- *Domain Parameter Group [MODP-3072, MODP-4096, MODP-6144, MODP-8192, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]*

Algorithm Functional Test
For each supported domain parameter group, the evaluator shall generate 10 test cases by generating the initiator and responder secret keys using random data, calculating the responder public key, and creating the shared secret. The resulting shared secrets shall be compared with those generated by a known-good implementation using the same inputs.

Validation Test
For each supported combination of the above parameters the evaluator shall generate 15 Diffie Hellman initiator/responder key pairs using the key generation function of a known-good implementation. For each set of key pairs, the evaluator shall modify five initiator private key values. The remaining key values are left unchanged (i.e., correct). To determine correctness, the evaluator shall confirm that the 15 shared secrets correspond as expected for both the modified and unmodified inputs.

Elliptic Curve Diffie-Hellman Key Agreement

Identifier	***Cryptographic Algorithm***	***Cryptographic Parameters***	***List of Standards***
ECDH	*Elliptic Curve Diffie-Hellman*	*Elliptic Curve [**selection:** P-384, P-521*	*NIST SP 800-56A Revision 3 (Section 5.7.1.2) [ECDH]* *NIST SP 800-186 (Section 3.2.1)*

			[NIST Curves]

To test the TOE's implementation of Elliptic Curve Diffie-Hellman Key Agreement, the evaluator shall perform the Algorithm Functional Test and Validation Test using the following input parameters:

- *Elliptic Curve [P-384, P-521]*

Algorithm Functional Test
For each supported Elliptic Curve the evaluator shall generate 10 test cases by generating the initiator and responder secret keys using random data, calculating the responder public key, and creating the shared secret. The resulting shared secrets shall be compared with those generated by a known-good implementation using the same inputs.

Validation Test
For each supported Elliptic Curve the evaluator shall generate 15 Diffie Hellman initiator/responder key pairs using the key generation function of a known-good implementation. For each set of key pairs, the evaluator shall modify five initiator private key values. The remaining key values are left unchanged (i.e., correct). To determine correctness, the evaluator shall confirm that the 15 shared secrets correspond as expected for the modified and unmodified values.
Curve25519 Key Establishment Schemes

Identifier	***Cryptographic Algorithm***	***Cryptographic Parameters***	***List of Standards***
ECDH-Ed	*ECDH with Montgomery Curves*	*Domain parameters approved for elliptic curves [curve25519]*	*RFC 7748 (Section 5) [ECDH-Ed]* *NIST SP 800-186 (Section 3.2.2) [Montgomery Curves]*

Curve25519 Key Establishment Schemes
The evaluator shall verify a TOE's implementation of the key agreement scheme using the following Function and Validity tests. These validation tests for each key agreement scheme verify that a TOE has implemented the components of the key agreement scheme according to the specification. These components include the calculation of the shared secret K and the hash of K.

Function Test
The Function test verifies the ability of the TOE to implement the key agreement schemes correctly. To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each supported key agreement role and hash function combination, the tester shall generate 10 sets of public keys. These keys are static, ephemeral or both depending on the scheme being tested.

The evaluator shall obtain the shared secret value K, and the hash of K.

The evaluator shall verify the correctness of the TSF's implementation of a given scheme by using a known good implementation to calculate the shared secret value K and compare the hash generated from this value.

Validation Test
The Validity test verifies the ability of the TOE to recognize another party's valid and invalid key agreement results. To conduct this test, the evaluator generates a set of 30 test vectors consisting of data sets including the evaluator's public keys and the TOE's public/private key pairs.

The evaluator shall inject an error in some of the test vectors to test that the TOE recognizes invalid key agreement results caused by the following fields being incorrect: the shared secret value K or the hash of K. At least two of the test vectors shall remain unmodified and therefore should result in valid key agreement results (they should pass).

The TOE shall use these modified test vectors to emulate the key agreement scheme using the corresponding parameters. The evaluator shall compare the TOE's results with the results using a known good implementation verifying that the TOE detects these errors.

FCS_CKM_EXT.7/UNLOCKED Cryptographic Key Establishment (When Unlocked)

This is an implementation-based component. Its inclusion in depends on whether the TOE implements one or more of the following features:

-

as described in Appendix A.3: Implementation-based Requirements.

FCS_CKM_EXT.7.1/UNLOCKED
The TSF shall derive shared cryptographic keys with input from multiple parties in accordance with specified cryptographic key agreement algorithms

[**selection**: *Cryptographic algorithm*] and specified cryptographic parameters [**selection**: *Cryptographic parameters*] that meet the following: [**selection**: *List of standards*] .

The following table provides the allowable choices for completion of the selection operations of FCS_CKM_EXT.7/UNLOCKED.

Table 10: Allowable choices for FCS_CKM_EXT.7

Identifier	Cryptographic algorithm	Cryptographic parameters	List of standards
KAS2	RSA	Modulus size [**selection**: *3072*, *4096*, *6144*, *8192*] bits	NIST SP 800-56B Revision 2 (Section 8.3) [KAS2]
DH	Finite Field Cryptography Diffie-Hellman	Static domain parameters approved for [**selection**: • *IKE Groups* [***selection**: MODP-3072, MODP-4096, MODP-6144, MODP-8192]* • *TLS Groups* [***selection**: ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]*]	NIST SP 800-56A Revision 3 (Section 5.7.1.1) [DH] [**selection**: *RFC 3526 [IKE groups], RFC 7919 [TLS groups]*]
ECDH	Elliptic Curve Diffie-Hellman	Elliptic Curve [**selection**: *P-384*, *P-521*]	NIST SP 800-56A Revision 3 (Section 5.7.1.2) [ECDH] NIST SP 800-186 (Section 3.2.1) [NIST Curves]

Application Note: This SFR is claimed if the TSF supports key agreement schemes as a method of key establishment for trusted channels. This will generally apply to all conformant TOEs, except in the rare (but possible) case where only key encapsulation is used.

All of the above algorithms with the selectable parameters are CNSA 1.0 compliant.

This SFR must be included in the ST if key agreement or transport is a service provided by the TOE to tenant software, or if they are used by the TOE itself to support or implement PP-specified security functionality.

Evaluation Activities

FCS_CKM_EXT.7/UNLOCKED
TSS
The evaluator shall ensure that the TSS documents that the security strength of the material contributed by the TOE is sufficient for the security strength of the key and the agreement method.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
The following tests are conditional based upon the selections made in the SFR. The evaluator shall perform the following test or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

KAS2

Identifier	*Cryptographic Algorithm*	*Cryptographic Parameters*	*List of Standards*
KAS2	*RSA*	*Modulus Size [**selection:** 3072, 4096, 6144, 8192] bits*	*NIST SP 800-56B Revision 2 (Section 8.3) [KAS2]*

To test the TOE's implementation of the of the KAS2 RSA Key Agreement scheme, the evaluator shall perform the Algorithm Functional Test and Validation Test using the following input parameters:

- *RSA Private key format [Basic, Prime Factor, Chinese Remainder Theorem]*

- *Modulo value [3072, 4096, 6144, 8192]*
- *Role [initiator, responder]*

The evaluator shall generate a test group (i.e. set of tests) for each parameter value of the above parameter type with the largest number of supported values. For example, if the TOE supports all five Modulo values, then the evaluator shall generate five test groups. Each of the above supported parameter values must be included in at least one test group.

Regardless of how many parameter values are supported, there must be at least two test groups.

Half of the test groups are designated as Algorithm Functional Tests (AFT) and the remainder are designated as Validation Tests (VAT). If there is an odd number of groups, then the extra group is designated randomly as either AFT or VAT.

Algorithm Functional Test
For each test group designated as AFT, the evaluator shall generate 10 test cases using random data (except for a fixed public exponent, if supported). The resulting shared secrets shall be compared with those generated by a known-good implementation using the same inputs.

Validation Test
For each test group designated as VAT, the evaluator shall generate 25 test cases are using random data (except for a fixed public exponent, if supported). Of the 25 test cases:

- *Two test cases must have a shared secret with a leading nibble of 0s,*
- *Two test cases have modified derived key material,*
- *Two test cases have modified tags, if key confirmation is supported,*
- *Two test cases have modified MACs, if key confirmation is supported, and*
- *The remaining test cases are not modified.*

To determine correctness, the evaluator shall confirm that the resulting 25 shared secrets correspond as expected for both the modified and unmodified values.

FFC Diffie-Hellman Key Agreement

Identifier	***Cryptographic Algorithm***	***Cryptographic Parameters***	***List of Standards***
DH	*Finite Field Cryptography Diffie-Hellman*	*Static domain parameters approved for [**selection:** IKE groups [**selection:** MODP-3072, MODP-4096, MODP-6144, MODP-8192], TLS groups [**selection:** ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]]]*	*NIST SP 800-56A Revision 3 (Section 5.7.1.1) [DH]* *[**selection:** RFC 3526 [IKE Groups], RFC 7919 [TLS Groups]]*

To test the TOE's implementation of FFC Diffie-Hellman Key Agreement, the evaluator shall perform the Algorithm Functional Test and Validation Test using the following input parameters:

- *Domain Parameter Group [MODP-3072, MODP-4096, MODP-6144, MODP-8192, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]*

Algorithm Functional Test
For each supported domain parameter group, the evaluator shall generate 10 test cases by generating the initiator and responder secret keys using random data, calculating the responder public key, and creating the shared secret. The resulting shared secrets shall be compared with those generated by a known-good implementation using the same inputs.

Validation Test
For each supported combination of the above parameters the evaluator shall generate 15 Diffie Hellman initiator/responder key pairs using the key generation function of a known-good implementation. For each set of key pairs, the evaluator shall modify five initiator private key values. The remaining key values are left unchanged (i.e., correct). To determine correctness, the evaluator shall confirm that the 15 shared secrets correspond as expected for both the modified and unmodified inputs.

Elliptic Curve Diffie-Hellman Key Agreement

Identifier	***Cryptographic Algorithm***	***Cryptographic Parameters***	***List of Standards***
ECDH	*Elliptic Curve Diffie-Hellman*	*Elliptic Curve [**selection:** P-384, P-521*	*NIST SP 800-56A Revision 3 (Section 5.7.1.2) [ECDH]* *NIST SP 800-186 (Section 3.2.1) [NIST Curves]*

To test the TOE's implementation of Elliptic Curve Diffie-Hellman Key Agreement, the evaluator shall perform the Algorithm Functional Test and Validation Test using the following input

parameters:

- *Elliptic Curve [P-384, P-521]*

Algorithm Functional Test
For each supported Elliptic Curve the evaluator shall generate 10 test cases by generating the initiator and responder secret keys using random data, calculating the responder public key, and creating the shared secret. The resulting shared secrets shall be compared with those generated by a known-good implementation using the same inputs.

Validation Test
For each supported Elliptic Curve the evaluator shall generate 15 Diffie Hellman initiator/responder key pairs using the key generation function of a known-good implementation. For each set of key pairs, the evaluator shall modify five initiator private key values. The remaining key values are left unchanged (i.e., correct). To determine correctness, the evaluator shall confirm that the 15 shared secrets correspond as expected for the modified and unmodified values.

FCS_CKM_EXT.8 Password-Based Key Derivation

FCS_CKM_EXT.8.1

The TSF shall perform password-based key derivation functions in accordance with a specified cryptographic algorithm [HMAC- [**selection**: *SHA-384*, *SHA-512*], with iteration count of [**assignment**: *number of iterations*] using a randomly generated salt of length [**assignment**: *equal to or greater than 128*] and output cryptographic key sizes [**selection**: *256*, *384*, *512*] bits that meet the following standard: [NIST SP 800-132 (Section 5.3) [PBKDF2]].

Application Note: NIST recommends a minimum “number of iterations” of 1000 but prefers the largest number feasible given performance constraints.

NIST recommends that the randomly generated portion of the salt have length of at least 128 bits and must be derived from Random Bit Generation.

If this SFR is claimed, then FCS_COP.1/KeyedHash and FCS_RBG.1 must also be claimed.

For CNSA 1.0 and 2.0 compliance, only SHA-384 or SHA-512 may be used.

Evaluation Activities

FCS_CKM_EXT.8
TSS
The evaluator must verify that the TSS documents that the selection of the keyed hash algorithm, iteration count, length of salt, and other mitigations are sufficient for the security strength of the key derived.

The evaluator shall examine the TSS to verify that the salt is generated in accordance with the relevant specification.

The evaluator shall examine the TSS to determine whether the TOE implements other mitigations against password-exhaustion attacks. Examples include the use of interface-based mitigations and secret salts.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
The following tests are conditional based upon the selections made in the SFR. The evaluator shall perform the following test or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

To test the TOE’s ability to derive cryptographic keys from a password using PBKDF2 the evaluator shall perform the Algorithm Functional Test using the following input parameters:

- *HMAC algorithms [SHA-384, SHA-512]*
- *Iteration count [1-10000000]*
- *Derived Key size [256, 384, 512] bits*
- *Password length [8-128] bytes*
- *Salt length [128-4096] bits in multiples of 8.*

Algorithm Functional Test

For each supported HMAC algorithm, the evaluator shall generate 50 test cases using supported values for the above parameters such that

- *All supported derived key sizes are tested at least 10 times,*
- *Iteration counts are random values between the supported minimum and maximum values, with the supported minimum and maximum tested at least once each,*

- *Passwords are random byte strings representing upper- and lower-case letters of random supported lengths such that the minimum and maximum lengths are tested at least once, and*
- *Salts are random values between the supported minimum and maximum lengths such that the supported minimum and maximum lengths are both tested at least once.*

The evaluator shall compare the resulting keys from each test case with keys derived using a known-good implementation with the same input parameters.

FCS_COP.1/AEAD Cryptographic Operation – Authenticated Encryption with Associated Data

FCS_COP.1.1/AEAD

The TSF shall perform [*authenticated encryption with associated data*] in accordance with a specified cryptographic algorithm [**selection**: *Cryptographic algorithm*] and cryptographic key sizes [**selection**: *Cryptographic key sizes*] that meet the following: [**selection**: *List of standards*]

The following table provides the allowable choices for completion of the selection operations of FCS_COP.1/AEAD.

Table 11: Allowable choices for FCS_COP.1/AEAD

Identifier	Cryptographic algorithm	Cryptographic key sizes	List of standards
AES-CCM	AES in CCM mode with unpredictable, non-repeating nonce, minimum size of 64 bits	256 bits	[**selection**: *ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197*] [AES] [**selection**: *ISO/IEC 19772:2020 (Clause 7), NIST SP 800-38C*] [CCM]
AES-GCM	AES in GCM mode with non-repeating IVs using [**selection**: *deterministic, RBG-based*], IV construction; the tag must be of length [**selection**: *96, 104, 112, 120, 128*] bits.	256 bits	[**selection**: *ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197*] [AES] [**selection**: *ISO/IEC 19772:2020 (Clause 10), NIST SP 800-38D*] [GCM]

Application Note: The use of 256-bit keys for AES encryption is required by CNSA 1.0 and 2.0.

Evaluation Activities

FCS_COP.1/AEAD
TSS
The evaluator shall examine the TSS to ensure that it describes the construction of any IVs, nonces, and tags in conformance with the relevant specifications.

If a CCM mode algorithm is selected, then the evaluator shall examine the TOE summary specification to confirm that it describes how the nonce is generated and that the same nonce is never reused to encrypt different plaintext pairs under the same key.

If a GCM mode algorithm is selected, then the evaluator shall examine the TOE summary specification to confirm that it describes how the IV is generated and that the same IV is never reused to encrypt different plaintext pairs under the same key. The evaluator shall also confirm that for each invocation of GCM, the length of the plaintext is at most (2 ^{32})-2 blocks.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
The following tests may require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products.

The following tests are conditional based upon the selections made in the SFR. The evaluator shall perform the following test or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

AES-CCM

Identifier	***Cryptographic Algorithm***	***Cryptographic Key Sizes***	***List of Standards***
AES-CCM	*AES in CCM mode with nonrepeating nonce, minimum size of 64 bits*	*256 bits*	*[**selection:** ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES]* *[**selection:** ISO/IEC 19772:2020 (Clause 7), NIST SP 800-38C] [CCM]*

To test the TOE's implementation of AES-CCM authenticated encryption functionality the evaluator shall perform the Algorithm Functional Tests described below using the following input parameters:

- *Key Size [256] bits*
- *Associated data size [0-65536] bits in increments of 8*
- *Payload size [0-256] bits in increments of 8*
- *IV/Nonce size [64-104] bits in increments of 8*
- *Tag size [32-128] bits in increments of 16*

Algorithm Functional Tests

Unless otherwise specified, the following tests should use random data, a tag size of 128 bits, IV/Nonce size of 104 bits, payload size of 256 bits, and associated data size of 256 bits. If any of these values are not supported, any supported value may be used. The evaluator shall compare the output from each test case against results generated by a known-good implementation with the same input parameters.

Variable Associated Data Test

For each claimed key size, and for each supported associated data size from 0 through 256 bits in increments of 8 bits, the TOE must be tested by encrypting 10 test cases using all random data. In addition, for each key size, the TOE must be tested by encrypting 10 cases with associated data lengths of 65536 bits, if supported.

Variable Payload Test

For each claimed key size, and for each supported payload size from 0 through 256 bits in increments of 8 bits, the TOE must be tested by encrypting 10 test cases using all random data.

Variable Nonce Test

For each claimed key size, and for each supported IV/Nonce size from 64 through 104 bits in increments of 8 bits, the TOE must be tested by encrypting 10 test cases using all random data.

Variable Tag Test

For each claimed key size, and for each supported tag size from 32 through 128 bits in increments of 16 bits, the TOE must be tested by encrypting 10 test cases using all random data.

Decryption Verification Test

For each claimed key size, for each supported associated data size from 0 through 256 bits in increments of 8 bits, for each supported payload size from 0 through 256 bits in increments of 8 bits, for each supported IV/Nonce size from 64 through 104 bits in increments of 8 bits, and for each supported tag size from 32 through 128 bits in increments of 16 bits, the TOE must be tested by decrypting 10 test cases using all random data.

AES-GCM

Identifier	***Cryptographic Algorithm***	***Cryptographic Key Sizes***	***List of Standards***
AES-GCM	*AES in GCM mode with nonrepeating IVs using [**selection:** deterministic, RBG-based] IV construction; the tag must be of length*	*256 bits*	*[**selection:** ISO/IEC 18033-3:2010*

	*[**selection:** 96, 104, 112, 120, or 128] bits.*		*(Subclause 5.2), FIPS PUB 197] [AES]* *[**selection:** ISO/IEC 19772:2020 (Clause 10), NIST SP 800-38D] [GCM]*

To test the TOE's implementation of AES-GCM authenticated encryption functionality the evaluator shall perform the Encryption Algorithm Functional Tests and Decryption Algorithm Functional Tests as described below using the following input parameters:

- *Key Size [256] bits*
- *Associated data size [0-65536] bits*
- *Payload size [0-65536] bits*
- *IV size [96] bits*
- *Tag size [96, 104, 112, 120, 128] bits*

Encryption Algorithm Functional Tests

The evaluator shall generate 15 test cases using random data for each combination of the above parameters as follows:

- *Each claimed key size,*
- *Each supported tag size,*
- *Four supported non-zero payload sizes, such that two are multiples of 128 bits and two are not multiples of 128 bits,*
- *Four supported non-zero associated data sizes, such that two are multiples of 128 bits and two are not multiples of 128 bits, and*
- *An associated data size of zero, if supported.*

Note that the IV size is always 96 bits.

The evaluator shall compare the output from each test case against results generated by a known- good implementation with the same input parameters.

Decryption Algorithm Functional Tests

The evaluator shall test the authenticated decrypt functionality of AES-GCM by supplying 15 test cases for the supported combinations of the parameters as described above. For each parameter combination the evaluator shall introduce an error into either the Ciphertext or the Tag such that approximately half of the cases are correct and half the cases contain errors.

FCS_COP.1/Hash Cryptographic Operation - Hashing

FCS_COP.1.1/Hash

The TSF shall perform [*cryptographic hashing*] in accordance with a specified cryptographic algorithm [**selection**: *SHA-256*, *SHA-384*, *SHA-512*, *SHA3-384*, *SHA3-512*] that meet the following: [**selection**: *ISO/IEC 10118-3:2018 [SHA, SHA3]*, *FIPS PUB 180-4 [SHA]*, *FIPS PUB 202 [SHA3]*].

Application Note: In accordance with CNSA 1.0 and 2.0:

- SHA-1 hash is no longer permitted to be used as a hash function,
- SHA3 hashes may be used only for internal hardware functionality such as boot integrity checks, and
- SHA-256 is permitted only for use as a PRF or MAC as part of a key derivation function, or as part of LMS or XMSS.

The hash selection should be consistent with the overall strength of the algorithm used for signature generation. For example, the TOE should choose SHA-384 for 3072-bit RSA, 4096-bit RSA, or ECC with P-384; and SHA-512 for ECC with P-521.

Evaluation Activities

FCS_COP.1/Hash
TSS
The evaluator shall examine the TSS to verify that if SHA-256 is selected, that it is being used only as a PRF or MAC step in a key derivation function or as part of LMS, and not as a hash algorithm.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
The following tests may require the developer to provide access to a test platform that provides

the evaluator with tools that are typically not found on factory products.

The following tests are conditional, based on the selections made in the SFR. The evaluator shall perform the following tests or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

SHA-256, SHA-384, SHA-512

To test the TOE's ability to generate hash digests using SHA2, the evaluator shall perform the Algorithm Functional Test, Monte Carlo Test, and Large Data Test for each claimed SHA2 algorithm.

Algorithm Functional Test

The evaluator shall generate a number of test cases equal to the block size of the hash (512 for SHA2-256; 1024 for the other SHA2 algorithms).

Each test case is to consist of random data of a random length between 0 and 65536 bits, or the largest size supported.

Monte Carlo Test

Monte Carlo tests begin with a single seed and run 100 iterations of the chained computation.

There are two versions of the Monte Carlo Test for SHA-1 and SHA-2. Either one is acceptable. For the standard Monte Carlo test the message hashed is always three times the length of the initial seed.

```
For j = 0 to 99
A = B = C = SEED
For i = 0 to 999
MSG = A || B || C
MD = SHA(MSG)
A = B
B = C
C = MD
Output MD
SEED = MD
```

For the alternate version of the Monte Carlo Test, the hashed message is always the same length as the seed.

```
INITIAL_SEED_LENGTH = LEN(SEED)
For j = 0 to 99
A = B = C = SEED
For i = 0 to 999
MSG = A || B || C
if LEN(MSG) >= INITIAL_SEED_LENGTH:
MSG = leftmost INITIAL_SEED_LENGTH bits of MSG
else:
MSG = MSG || INITIAL_SEED_LENGTH - LEN(MSG) 0 bits
MD = SHA(MSG)
A = B
B = C
C = MD
Output MD
SEED = MD
```

The evaluator shall compare the output against results generated by a known good implementation with the same input.

Large Data Test

The implementation must be tested against one test case each on large data messages of 1GB, 2GB, 4GB, and 8GB of data as supported. The data need not be random. It may, for example, consist of a repeated pattern of 64 bits.

The evaluator shall compare the output against results generated by a known good implementation with the same input.

SHA3-384, SHA3-512

To test the TOE's ability to generate hash digests using SHA3 the evaluator shall perform the Algorithm Functional Test, Monte Carlo Test, and Large Data Tests for each claimed SHA3 algorithm.

Algorithm Functional Test

Generate a test case consisting of random data for every message length from 0 bits (or the smallest supported message size) to rate bits, where rate equals

- *832 for SHA3-384 and*
- *576 for SHA3-512.*

Additionally, generate tests cases of random data for messages of every multiple of (rate+1) bits starting at length rate, and continuing until 65535 is exceeded.

The evaluator shall compare the output against results generated by a known good implementation with the same input.

Monte Carlo Test

Monte Carlo tests begin with a single seed and run 100 iterations of the chained computation.

For this Monte Carlo Test, the hashed message is always the same length as the seed.

```
MD[0] = SEED
INITIAL_SEED_LENGTH = LEN(SEED)
For 100 iterations
For i = 1 to 1000
MSG = MD[i-1];
if LEN(MSG) >= INITIAL_SEED_LENGTH:
MSG = leftmost INITIAL_SEED_LENGTH bits of MSG
else:
MSG = MSG || INITIAL_SEED_LENGTH - LEN(MSG) 0 bits
MD[i] = SHA3(MSG)
MD[0] = MD[1000]
Output MD[0]
```

The evaluator shall compare the output against results generated by a known good implementation with the same input.

Large Data Test

The implementation must be tested against one test case each on large data messages of 1GB, 2GB, 4GB, and 8GB of data as supported. The data need not be random. It may, for example, consist of a repeated pattern of 64 bits.

The evaluator shall compare the output against results generated by a known good implementation with the same input.

FCS_COP.1/KeyedHash Cryptographic Operation - Keyed Hash

FCS_COP.1.1/KeyedHash

The TSF shall perform [*keyed hash message authentication*] in accordance with a specified cryptographic algorithm [**selection**: *Keyed Hash Algorithm*] and cryptographic key sizes [**selection**: *Cryptographic Key Sizes*] that meet the following: [**selection**: *List of Standards*]

The following table provides the allowable choices for completion of the selection operations of FCS_COP.1/KeyedHash.

Table 12: Allowable choices for FCS_COP.1/KeyedHash

Keyed Hash Algorithm	Cryptographic Key Sizes	List of Standards
HMAC-SHA-256	256 bits	[**selection**: *ISO/IEC 9797-2:2021 (Section 7 “MAC Algorithm 2”) , FIPS PUB 198-1*]
HMAC-SHA-384	[**selection**: *384 (ISO, FIPS), 256 (FIPS)*] bits	[**selection**: *ISO/IEC 9797-2:2021 (Section 7 “MAC Algorithm 2”) , FIPS PUB 198-1*]
HMAC-SHA-512	[**selection**: *512 (ISO, FIPS), 384 (FIPS), 256 (FIPS)*] bits	[**selection**: *ISO/IEC 9797-2:2021 (Section 7 “MAC Algorithm 2”) , FIPS PUB 198-1*]

Application Note: The intent of this requirement is to specify the keyed-hash message authentication function used for key establishment purposes for the various cryptographic protocols used by the OS (e.g., trusted channel). The hash selection must support the message digest size selection. The hash selection should be consistent with the overall strength of the algorithm used for FCS_COP.1/Hash.

In accordance with CNSA 1.0 and 2.0, HMAC-SHA-256 may be used only as a PRF or MAC step in a key derivation function.

Evaluation Activities

FCS_COP.1/KeyedHash
TSS
The evaluator shall examine the TSS to ensure that the size of the key is sufficient for the desired security strength of the output.

The evaluator shall examine the TSS to verify that if HMAC-SHA-256 is selected, that it is being used only as a PRF or MAC step in a key derivation function.

Guidance

There are no additional Guidance evaluation activities for this component.

Tests
The following tests are conditional based on the selections made in the SFR. The evaluator shall perform the following tests or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

HMAC

Keyed Hash Algorithm	***Cryptographic Key Sizes***	***List of Standards***
HMAC-SHA-256	*256 bits*	*[**selection:** ISO/IEC 9797-2:2021 (Section 7 "MAC Algorithm 2"), FIPS PUB 198-1]*
HMAC-SHA-384	*[**selection:** (ISO, FIPS) 384, (FIPS) 256] bits*	*[**selection:** ISO/IEC 9797-2:2021 (Section 7 "MAC Algorithm 2"), FIPS PUB 198-1]*
HMAC-SHA-512	*[**selection:** (ISO, FIPS) 512, (FIPS) 384, 256] bits*	*[**selection:** ISO/IEC 9797-2:2021 (Section 7 "MAC Algorithm 2"), FIPS PUB 198-1]*

To test the TOE's ability to generate keyed hashes using HMAC the evaluator shall perform the Algorithm Functional Test for each combination of claimed HMAC algorithm the following parameters:

- *Hash function [SHA-256, SHA-384, SHA-512]*
- *Key length [8-65536] bits by 8s*
- *MAC length [32-[digest size of hash function (256, 384, 512)]] bits*

Algorithm Functional Test

For each supported Hash function the evaluator shall generate 150 test cases using random input messages of 128 bits, random supported key lengths, random keys, and random supported MAC lengths such that across the 150 test cases:

- *The key length includes the minimum, the maximum, a key length equal to the block size, and key lengths that are both larger and smaller than the block size.*
- *The MAC size includes the minimum, the maximum, and two other random values.*

The evaluator shall compare the output against results generated by a known good implementation with the same input.

FCS_COP.1/KeyEncap Cryptographic Operation - Key Encapsulation

This is a selection-based component. Its inclusion depends upon selection from FCS_CKM.2.1.

FCS_COP.1.1/KeyEncap

The TSF shall perform [key encapsulation] in accordance with a specified cryptographic algorithm [**selection**: *Cryptographic Algorithm*] and cryptographic key sizes [**selection**: *Cryptographic Key Sizes*] that meet the following: [**selection**: *List of Standards*]

The following table provides the allowable choices for completion of the selection operations of FCS_COP.1/KeyEncap.

Table 13: Allowable choices for FCS_COP.1/KeyEncap

Identifier	Cryptographic Algorithm	Cryptographic Key Sizes	List of Standards
ML-KEM	ML-KEM	Parameter set = ML-KEM-1024	NIST FIPS 203

Application Note: For this PP, the only anticipated use of key encapsulation is the use of ML-KEM as part of key establishment for trusted communications.

Evaluation Activities

FCS_COP.1/KeyEncap
TSS
The evaluator shall ensure that the TSS documents that the selection of the key size is sufficient for the security strength of the key encapsulated.

The evaluator shall examine the TSS to verify that any one-time values such as nonces or masks are constructed and used in accordance with the relevant standards.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
The following test may require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products.

The following tests are conditional based upon the selections made in the SFR. The evaluator shall perform the following test or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

ML-KEM Key Encapsulation

Identifier	***Cryptographic Algorithm***	***Cryptographic Key Sizes***	***List of Standards***
ML-KEM	*ML-KEM*	*Parameter set = ML-KEM-1024*	*NIST FIPS PUB 203*

To test the TOE's implementation of ML-KEM key encapsulation/decapsulation, the evaluator shall perform the Encapsulation Test and the Decapsulation Test using the following input parameters:

- *Encapsulation Parameters:*
  - *Parameter set [ML-KEM-1024]*
  - *Previously generated encapsulation key (ek)*
  - *Random value (m) [32 bytes]*
- *Decapsulation Parameters:*
  - *Parameter set [ML-KEM-1024]*
  - *Previously generated decapsulation key (dk)*
  - *Previously generated ciphertext (c) [32 bytes]*

Encapsulation Test

For each supported parameter set the evaluator shall generate 25 test cases consisting of an encapsulation key ek and random value m. For each test case the valuator shall require the implementation under test to generate the corresponding shared secret k and ciphertext c. To determine correctness, the evaluator shall compare the resulting values with those generated using a known-good implementation using the same inputs.

Encapsulation Key Check (if supported)

The evaluator shall generate 10 encapsulation keys such that:

- *Five of the encapsulation keys are valid, and*
- *Five of the encapsulation keys are modified such that a value in the noisy linear system is encoded into the key as a value greater than Q.*

The evaluator shall invoke the TOE's Encapsulation Key Check functionality to determine the validity of the 10 keys. The unmodified keys should be determined valid, and the modified keys should be determined invalid.

Decapsulation Key Check (if supported)

The evaluator shall generate 10 decapsulation keys such that:

- *Five of the decapsulation keys are valid, and*
- *Five of the decapsulation keys are modified such that the concatenated values ek||H(ek) will no longer match by modifying H(ek) to be a different value.*

The evaluator shall invoke the TOE's Decapsulation Key Check functionality to determine the validity of the 10 keys. The unmodified keys should be determined valid, and the modified keys should be determined invalid.

Decapsulation Test

For each supported parameter set the evaluator shall use a single previously generated decapsulation key dkand generate 10 test cases consisting of valid and invalid ciphertexts c. For each test case the evaluator shall require the implementation under test to generate the corresponding shared secret kwhether or not the ciphertext is valid. To determine correctness, the evaluator shall compare the resulting values with those generated using a known-good implementation using the same inputs.

FCS_COP.1/KeyWrap Cryptographic Operation - Key Wrapping

FCS_COP.1.1/KeyWrap

The TSF shall perform [*key wrapping*] in accordance with a specified cryptographic algorithm [**selection**: *Cryptographic Algorithm*] and

cryptographic key sizes [**selection**: *Cryptographic Key Sizes*] that meet the following: [**selection**: *List of Standards*]

The following table provides the allowable choices for completion of the selection operations of FCS_COP.1/KeyWrap.

Table 14: Allowable choices for FCS_COP.1/KeyWrap

Identifier	Cryptographic Algorithm	Cryptographic Key Sizes	List of Standards
AES-KW	AES in KW mode	256 bits	[**selection**: *ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197*] [AES] [**selection**: *ISO/IEC 19772:2020 (clause 6), NIST SP 800-38F (Section 6.2)*] [KW mode]
AES-KWP	AES in KWP mode	256 bits	[**selection**: *ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197*] [AES] NIST SP 800-38F (Section 6.3) [KWP mode]
AES-CCM	AES in CCM mode with unpredictable, non-repeating nonce, minimum size of 64 bits	256 bits	[**selection**: *ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197*] [AES] [**selection**: *ISO/IEC 19772:2020 (Clause 7), NIST SP 800-38C*] [CCM]
AES-GCM	AES in GCM mode with non-repeating IVs using [**selection**: *deterministic*, *RBG-based*], IV construction; the tag must be of length [**selection**: *96*, *104*, *112*, *120*, *128*] bits.	256 bits	[**selection**: *ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197*] [AES] [**selection**: *ISO/IEC 19772:2020 (Clause 10), NIST SP 800-38D*] [GCM]

Application Note: NIST 800-57p1rev5 sec. 5.6.2 specifies that the size of key used to protect the key being transported should be at least the security strength of the key it is protecting.

Evaluation Activities

FCS_COP.1/KeyWrap

TSS

The evaluator shall ensure that the TSS documents that the selection of the key size is sufficient for the security strength of the key wrapped.

The evaluator shall examine the TSS to ensure that it describes the construction of any IVs, nonces, and MACs in conformance with the relevant specifications.

Guidance

There are no additional Guidance evaluation activities for this component.

Tests

For tests of AES-GCM and AES-CCM, see testing for FCS_COP.1/AEAD.

The following tests are conditional based upon the selections made in the SFR. The evaluator shall perform the following test or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

AES-KW

Identifier	***Cryptographic Algorithm***	***Cryptographic Key Sizes***	***List of Standards***
AES-KW	*AES in KW mode*	*256 bits*	*[**selection:** ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES]* *[**selection:** ISO/IEC 19772:2020 (clause 6), NIST SP 800-38F (Section 6.2)] [KW mode]*

To test the TOE's ability to wrap keys using AES in Key Wrap mode the evaluator shall perform the Algorithm Functional Tests using the following input parameters:

- *Key size [256] bits*
- *Keyword cipher type [cipher, inverse]*
- *Payload sizes [128-4096] bits by 64s*

Algorithm Functional Test

The evaluator shall generate 100 encryption test cases using random data for each combination of claimed key size, keyword cipher type, and six supported payload sizes such that the payload sizes include the minimum, the maximum, two that are divisible by 128, and two that are not divisible by 128.

The results shall be compared with those generated by a known-good implementation using the same inputs.

The evaluator shall generate 100 decryption test cases using the same parameters as above, but with 20 of each 100 test cases having modified ciphertext to produce an incorrect result. To determine correctness, the evaluator shall confirm that the results correspond as expected for both the modified and unmodified values.

AES-KWP

Identifier	***Cryptographic Algorithm***	***Cryptographic Key Sizes***	***List of Standards***
AES-KWP	*AES in KWP mode*	*256 bits*	*[**selection:** ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES]* *NIST SP 800-38F (Section 6.3) [KWP mode]*

To test the TOE's ability to wrap keys using AES in Key Wrap with Padding mode with padding the evaluator shall perform the Algorithm Functional Tests using the following input parameters:

- *Key size [256] bits*
- *Keyword cipher type [cipher, inverse]*
- *Payload sizes [8-4096] bits by 8s*

Algorithm Functional Test

The evaluator shall generate 100 encryption test cases using random data for each combination of claimed key size, keyword cipher type, and six supported payload sizes such that the payload sizes include the minimum, the maximum, two that are divisible by 128, and two that are not divisible by 128.

The results shall be compared with those generated by a known-good implementation using the same inputs.

The evaluator shall generate 100 decryption test cases using the same parameters as above, but

with 20 of each 100 test cases having modified ciphertext to produce an incorrect result. To determine correctness, the evaluator shall confirm that the results correspond as expected for both the modified and unmodified values.

FCS_COP.1/SigGen Cryptographic Operation - Signature Generation

FCS_COP.1.1/SigGen

The TSF shall perform [*digital signature generation*] in accordance with a specified cryptographic algorithm [**selection**: *Cryptographic Algorithm*] and cryptographic key sizes [**selection**: *Cryptographic Key Sizes*] that meet the following: [**selection**: *List of Standards*]

The following table provides the allowable choices for completion of the selection operations in FCS_COP.1/SigGen.

Table 15: Allowable choices for FCS_COP.1/SigGen

Identifier	Cryptographic Algorithm	Cryptographic Key Sizes	List of Standards
RSA-PKCS	RSASSA-PKCS1-v1_5	Modulus of size [**selection**: *3072*, *4096*, *6144*, *8192*] bits, hash [**selection**: *SHA-384*, *SHA-512*]	RFC 8017 (Section 8.2) [PKCS #1 v2.2] FIPS PUB 186-5 (Section 5.4) [RSASSA-PKCS1-v1_5]
RSA-PSS	RSASSA-PSS	Modulus of size [**selection**: *3072*, *4096*, *6144*, *8192*] bits, hash [**selection**: *SHA-384*, *SHA-512*], Salt Length () such that [**assignment**: *0 ≤ sLen ≤ hLen (Hash Output Length)*] and Mask Generation Function = MGF1	RFC 8017 (Section 8.1) [PKCS#1 v2.2] FIPS PUB 186-5 (Section 5.4) [RSASSA-PSS]
ECDSA	ECDSA	Elliptic Curve [**selection**: *P-384*, *P-521*], per-message secret number generation [**selection**: *extra random bits*, *rejection sampling*, *deterministic*] and hash function using [**selection**: *SHA-384*, *SHA-512*]	[**selection**: *ISO/IEC 14888-3:2018 (Subclause 6.6)*, *FIPS PUB 186-5 (Sections 6.3.1, 6.4.1*] [ECDSA] NIST SP-800 186 (Section 4) [NIST Curves]
LMS	LMS	Private key size =undefined Winternitz parameter =undefined Tree height = [**selection**: *5*, *10*, *15*, *20*, *25*]	RFC 8554 [LMS] NIST SP 800-208 [parameters]
ML-DSA	ML-DSA Signature Generation	Parameter set = ML-DSA-87	NIST FIPS 204 (Section 5.2)
XMSS	XMSS	Private key size =undefined Tree height = [**selection**: *10*, *16*, *20*]	RFC 8391 [XMSS] NIST SP 800-208 [parameters]

Application Note: The ST author should choose the algorithm implemented to perform digital signatures; if more than one algorithm is available, this requirement should be iterated to specify the functionality. For the algorithm chosen, the ST author should make the appropriate assignments and selections to specify the parameters that are implemented for that algorithm.

Evaluation Activities

FCS_COP.1/SigGen

TSS

The evaluator shall examine the TSS and verify that any hash function is the appropriate security strength for the signing algorithm.

The evaluator shall examine the TSS to verify that any one-time values such as nonces or masks are constructed and used in accordance with the relevant standards.

The evaluator shall examine the TSS to verify that the TOE has appropriate measures in place to ensure that hash-based signature algorithms do not reuse private keys.

Guidance

There are no additional Guidance evaluation activities for this component.

Tests

The following tests are conditional based on the selections made in the SFR. The evaluator shall perform the following tests or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

RSA-PKCS Signature Generation

Identifier	***Cryptographic Algorithm Parameters***	***Cryptographic Key Sizes***	***List of Standards***
RSA-PKCS	*RSASSA-PKCS1-v1_5*	*Modulus of size [**selection:** 3072, 4096, 6144, 8192] bits, hash [**selection:** SHA-384, SHA-512]*	*RFC 8017 (Section 8.2) [PKCS #1 v2.2]* *NIST FIPS PUB 186-5 (Section 5.4) [RSASSA-PKCS1-v1_5]*

To test the TOE's ability to perform RSA Digital Signature Generation using PKCS1-v1,5 signature type, the evaluator shall perform the Generated Data Test using the following input parameters:

- *Modulus size [3072, 4096, 6144, 8192] bits*
- *Hash algorithm [SHA-384, SHA-512]*

Generated Data Test

For each supported combination of the above parameters, the evaluator shall cause the TOE to generate three test cases using random data. The evaluator shall compare the results against those from a known good implementation.

RSA-PSS Signature Generation

Identifier	***Cryptographic Algorithm Parameters***	***Cryptographic Key Sizes***	***List of Standards***
RSA-PSS	*RSASSA-PSS*	*Modulus of size [**selection:** 3072, 4096, 6144, 8192] bits, hash [**selection:** SHA-384, SHA-512], Salt Length (sLen) such that [**assignment:** 0 ≤ sLen ≤ hLen (Hash Output Length)] and Mask Generation Function = MGF1*	*RFC 8017 (Section 8.2) [PKCS #1 v2.2]* *NIST FIPS PUB 186-5 (Section 5.4) [RSASSA-PSS]*

To test the TOE's ability to perform RSA Digital Signature Generation using PSS signature type, the evaluator shall perform the Generated Data Test using the following input parameters:

- *Modulus size [3072, 4096, 6144, 8192] bits*
- *Hash algorithm [SHA-384, SHA-512]*
- *Salt length [Fixed based on implementation]*
- *Mask function [MGF1]*

Generated Data Test

For each supported combination of the above parameters, the evaluator shall cause the TOE to generate three test cases using random data. The evaluator shall compare the results against those from a known good implementation.

ECDSA Signature Generation

Identifier	Cryptographic Algorithm Parameters	Cryptographic Key Sizes	List of Standards
ECDSA	ECDSA	Elliptic Curve [**selection:** P-384, P-521], per-message secret number generation [**selection:** extra random bits, rejection sampling, deterministic] and hash function using [**selection:** SHA-384, SHA-512]	[**selection:** ISO/IEC 14888-3:2018 (Subclause 6.6), NIST FIPS PUB 186-5 (Sections 6.3.1, 6.4.1] [ECDSA] NIST SP-800 186 (Section 4) [NIST Curves]

To test the TOE's ability to perform ECDSA Digital Signature Generation using extra random bits or rejection sampling for secret number generation, the evaluator shall perform the Algorithm Functional Test using the following input parameters:

- *Elliptic Curve [P-384, P-521]*
- *Hash algorithm [SHA-384, SHA-512]*

To test the TOE's ability to perform ECDSA Digital Signature Generation using deterministic secret number generation, the evaluator shall perform the Algorithm Functional Test using the following input parameters:

- *Elliptic Curve [P-384, P-521]*
- *Hash algorithm [SHA-384, SHA-512]*

Algorithm Functional Test

For each supported combination of the above parameters, the evaluator shall cause the TOE to generate 10 test cases using random data. The evaluator shall compare the results against those from a known good implementation.

LMS Signature Generation

Identifier	Cryptographic Algorithm Parameters	Cryptographic Key Sizes	List of Standards
LMS	LMS	Private key size = [**selection:** 192 bits with [**selection:** SHA256/192, SHAKE256/192], 256 bits with [**selection:** SHA-256, SHAKE256]], Winternitz parameter = [**selection:** 1, 2, 4, 8], and tree height = [**selection:** 5, 10, 15, 20, 25]	RFC 8554 [LMS] NIST SP 800-208 [parameters]

To test the TOE's ability to generate cryptographic digital signature using LMS, the evaluator shall perform the Algorithm Functional Test using the following input parameters:

- *Hash algorithm [SHA-256/192, SHAKE256/192, SHA-256, SHAKE256]*
- *Winternitz [1, 2, 4, 8]*
- *Tree height [5, 10, 15, 20, 25]*

Algorithm Functional Test

For each supported combination of the above parameters, the evaluator shall generate 10 signatures. The evaluator shall verify the correctness of the implementation by comparing values generated by the TOE with those generated by a known good implementation using the same input parameters.

ML-DSA Signature Generation

Identifier	Cryptographic Algorithm Parameters	Cryptographic Key Sizes	List of Standards
ML-DSA	ML-DSA SigGen	Parameter set = ML-DSA-87	NIST FIPS PUB 204 (Section 5.2)

To test the TOE's ability to generate digital signatures using ML-DSA, the evaluator shall perform the Algorithm Functional Test using the following input parameters:

- *Parameter set [ML-DSA-87]*
- *Seed [32 random bytes] (for non-deterministic signature testing), or*
- *Seed [32 zero bytes] (for deterministic signature testing)*
- *Message to sign [8-65535] bytes*
- *Mu value (if generated externally)*
- *Previously generated private key (sk)*
- *Context (for external interface testing)*

Algorithm Functional Test

For each combination of supported parameter set and capabilities, the evaluator shall require the implementation under test to generate 15 signatures pairs using 15 different randomly generated 32-byte seed values. To determine correctness, the evaluator shall compare the resulting key pairs with those generated using a known good implementation using the same inputs.

Known Answer Test for Rejection Cases

For each supported parameter set, the evaluator shall cause the TOE to generate signatures using the data below and a deterministic seed of all 0's. Correctness is determined by comparing the hash of the resulting signature with the hash in the fourth row for each corresponding test case below.

The test values are defined as follows:

- *Seed is the seed to generate the key pair (pk, sk)*
- *Hash of keys is computed by SHA-256(pk||sk)*
- *Message is the message to be signed*
- *Hash of sig is computed by SHA-256(sig)*

ML-DSA-87 Test Cases for Rejection Cases

```
Test case 87-RC-01
Seed:                   E4F5AFCF697E0EC3C1BDEB66FAA903221E803902F9C3F716E1056A63D77DC250
Hash of Keys:   61618E8DDA6998072C8EB36974E03880D741CAF0BD523356DFC161E7C9E63934
Message:                F4F1C05004D5B946F69EAFE104C4020519086ADDB9582A20FDE887D13DFC36B1
Hash of sig:    B584E38FA442FC3C81A147D4BDBF058D73C822CAF5CA4C06B0110867F60A8001

Test case 87-RC-02
Seed:                   8B828D871254D6C57384A8E7025AA3F7160CAD1D2C754499DF3844426062C3DD
Hash of Keys:   BB64481317D6C0DBAD20C0C7EF11078AD54E5D574F4A07652115A95F77C655FA
Message:                0F9409C5A4930C25B83FC5B77FDB5BB49C75372DE724D9C1A77DB700CF0CF154
Hash of sig:    F86B49BE9DEB2B209BDEB4E922E5939E92D38E562C44BB09AFBD67323C345192

Test case 87-RC-03
Seed:                   E693D282CACB8CE65FD4D108DA7A373F097F0AA9713550BE242AAD5BD3E2E452
Hash of Keys:   B0BEAF56713A69BD4AB2CBEE006FA5001E7B41F3AE541E05F088933AA0CC78DF
Message:                24DABB9D57ADEBD560ED65D9451C5106D437061708F849BA53F3543CDF9AAAE0
Hash of sig:    DBF65CEFF9F96A74AAF6F3AB27B043231BE6AA04FBA2EEC987A24A00BDD6A08E

Test case 87-RC-04
Seed:                   4002163EB8EED01A8E0919BA8C07D291341EDCAE25B02B9779A2CFFE50561AF0
Hash of Keys:   FED1BE685C20ECB322FC40D41DEE7E0E98D0409FBF989CAE71B8AD2D58AD645E
Message:                EE316BB5EBED53325B4A55571C60657B53E353B51B831F4A0BBB28107EBA4BA8
Hash of sig:    3BE9B5545FDCED92547B3409C83B3312CCB5792A8EC3A4DA63BA692C79BEF17C

Test case 87-RC-05
Seed:                   9C7AD524F65854C27E565BCEDF8E86D650F13A40D0448F9AE10C05F10F777120
Hash of Keys:   0EA872CA5A4BEA94F4E8EF7ED31800727899A51059FDEE111E5CB15F0233B534
Message:                CE09831294AA96CAF684B9E667947B021C57B24C138EC7D4DA270694C82F2E08
Hash of sig:    3B9526CEE6587F2418BFE603ADB0F7DF0D69EBA31C9F9F005C60C993945EBD33

Test case 87-RC-06
Seed:                   2EB7676D4A28700DA7772A7A035EB495CAA6F842352A74824EF5FD891BC38B2A
Hash of Keys:   D5B73703A1DDC5BCB0D14AE39B193A25D6ADA6535827973181ADB0BE70435A5B
Message:                C2B3A0AC483A5517682285C205974B2A506946448A8F7D3E1934C155EFDFE922
Hash of sig:    375D598704B722C8A1FEF1626FD7738A532C06329AA4217357460E3B729660F8

Test case 87-RC-07
Seed:                   E4E80CCE8B26DF1B02B99949851EE2F907FE4F0CC34790352C76D5D91634D073
Hash of Keys:   84B7E61684A12698400B09EA332EA3C4FBCFA47FE37FD6AE725CBC5FA8A99D3F
Message:                89E6AB43C9CB1CC59C3986D53217A558357E62102A26F666F2B64CD1DBB7A536
Hash of sig:    7C4AABD163CAEF8F6EBFDA3E3EEBC0A9604675B0E991ABAFD284F1AE8BA07B2A

Test case 87-RC-08
Seed:                   5787262B803499223D4E5A8C1EE572E89F7A69B359B3F8505355B0BDEAB95E5C
Hash of Keys:   85AE1DE605A7B479C02730BF4B7DD6D0FD8FFE5C980893CA6DAD00BD8BD1CE68
Message:                D3230C4E061964BBFB17702432D5D36FC1EB3D1068F8CCAA84044776E3B5CC55
Hash of sig:    D3ABE460EE2DD9595F413CFE2780A319E4E4DFD6592995298A7AB0B82A5E2815

Test case 87-RC-09
Seed:                   CE099B99330537DD153052243FC32ACAD509A126AB9824102585858567D410D79
Hash of Keys:   E04A9F15EDF8F078EB336CE624249EF2A8EDF2CDBF6A8276E9F5E92ED9B0BAE8
Message:                0035931762665F561A1B22176567E3B10FDE2441521F77030733A8E39312EEEE
Hash of sig:    3EEF413CB5EB179896ECA172D0DBFB9B251545DC561D61580BD5BBC8B6D734E1

Test case 87-RC-10
Seed:                   FC8F2929878CBD81E1CCC23913F290380120C043A4A8A251AEEBF09705B8E590
Hash of Keys:   7E2ECCA86F532E8E8092FEBB6E0007F92E7909AD2BCBE2E02AB375DAC9969E5E
Message:                D3C28875D2671C0EF23BFDC8869E8ECF8868D3F0561C3134D254F7479D0CE0E5
Hash of sig:    EB69A908EDCC04320A0B61AD57E21B044465F2037698636B64229CF2DB259789
```

XMSS Signature Generation

Identifier	***Cryptographic Algorithm Parameters***	***Cryptographic Key Sizes***	***List of Standards***
XMSS	*XMSS*	*Private key size = [**selection:** 192 bits with [**selection:** SHA256/192, SHAKE256/192], 256 bits with [**selection:** SHA-256, SHAKE256]], and tree height = [**selection:** 10, 16, 20]*	*RFC 8391 [XMSS]* *NIST SP 800-208 [parameters]*

To test the TOE's ability to generate digital signatures using XMSS, the evaluator shall perform the XMSS Key Generation Test using the following input parameters:

- *Hash algorithm [SHA-256/192, SHAKE256/192, SHA-256, SHAKE256]*
- *Tree height [10, 16, 20]*

XMSS Key Generation Test

For each supported combination of the above parameters, the evaluator shall generate 10 signatures. The evaluator shall verify the correctness of the implementation by comparing values generated by the TOE with those generated by a known-good implementation using the same input parameters.

Known Answer Test for Large Number of Rejection Cases (Total Rejection Count)

For each supported parameter set, the evaluator shall cause the TOE to generate signatures using the data below and a deterministic seed of all 0's. Correctness is determined by comparing the hash of the resulting signature with the hash in the fourth row of the corresponding test case below.

ML-DSA-87 Test Cases for Total Rejection Count

```
Test case 87-LN-01
Seed:                   98B6298051D92BF37293C93C97370747BF527B87B71F6C4264182F45155ADE4C
Hash of Keys:   04A135B5C9B7020332C7B16E7108E8FF7FC1EAE1C23C5FA0B5D5CED0FEEE7424
Message:                D7B0341269259083ABF3C8DC47559A19D57669B4486E0224F376DC43E577A3D8
Hash of sig:    58D72D76EC0FB65BFB9893C4479366B79DD7B8B7577E4291D13514FCC76C26DD

Test case 87-LN-02
Seed:                   DFB5BDD90F58571DCA962426C623F13D046BBE814D183886AC90D143EAD725A7
Hash of Keys:   2B6AB8CFCCCC41F759CAF01932E9413F5DC6D949BC827F739866929683FB155E
Message:                21005DB2B583CC826A9684BFFD0EE00AB97E0479FE4A1D266699337540145778
Hash of sig:    C93EA34E00FFFFC3ECEA072D5FB038A83B5539CAF7B831AEDCFA785E50B3CA5E

Test case 87-LN-03
Seed:                   5AD414E0DD0EF2FE685F342871875FDF06F503717A86C3B3466565ADD2096417
Hash of Keys:   BD9C2D52F3FC78DB17E682DA2E78947ECFC0898333838D60C892700B2B0DDA9F
Message:                29139C279816B25F2D6BB52C8247D163544F7BA332C3CF63359B9E23FBC56515
Hash of sig:    DB4BE2DE19FB40437BDB7E9B6578D665DB05B4E88C16907DF4546EBA9BE03AEA

Test case 87-LN-04
Seed:                   484DD2F406A4D15F49A91AD5FC3BDC1D0FF253622EB68F83D6E1C870D0E89E29
Hash of Keys:   A719DC9A77C91C46295555C2353BA0CBEA513DA9A92A5C34D2E949EFF46A12D8
Message:                6AD6E959F0EA60126364FB7C95FA71133F246A9265A11B4965EE78AB0CB5AF0E
Hash of sig:    5050D7A665074EC63D9F3966C1F01A1BFB18F9E83AE0B09F838BC1E2342ED6F4

Test case 87-LN-05
Seed:                   B25C1816F82D59940D5CB829BAC364AAD013C4C16415CE1CF6DCC2F15199B391
Hash of Keys:   ADBB2CD43F222640BD9FF4E61C80E63853E8DC1F759C581B7447C9C166EAA38E
Message:                824E47322895BFFE37B6B4AFC41CF6115C07EEC0C24EB81076C87A1B01AE8617
Hash of sig:    667ADA46073BC69D64DC47BB9A76DD0D78302E7415D87D5E816B05FB95F9E84D

Test case 87-LN-06
Seed:                   B2CE72B3560AF07E06465881F56ADA00262BA708D87B73F39E04E310F3B8A3E9
Hash of Keys:   FD9C4AC53AE803242A62DF933B8E8BAD6CE5207AC4A73683B6D9383B5E70B17A
Message:                A1501CC84C917E0D2D7C27C2AC382220BD8FFFE807DB38E37A9E429EC2781911
Hash of sig:    779553B195E11558EE59EF3942F5F6B446A2144600D1F4F50B300C6C56504760

Test case 87-LN-07
Seed:                   AB01D0E591B7DDCD3C03395AED808FA2763C0A486D44119D621BE0FD0B022B25
Hash of Keys:   93B6ADE34F78A4ADB36B2F6D2C51DB793E659E1243E80488AE1C03B65125D6D7
Message:                8DE8122D89D15FE84A4C34F6B59B2C4B11F33B6A053154D199B634F557FDF5F6
Hash of sig:    0483045999A79B583F403DB96A736F0F0B24E2DFBC4E5CFA9B50E3D910786F07

Test case 87-LN-08
Seed:                   15D60D3693762F82C9AC1DCB0576936651AC81D863842EDB91109C8EE83AE705
Hash of Keys:   2DF544E2E939AA717741C2437288FAEB308DEB8FF37A2652FAE34BAE8B84D779
Message:                F05946A6113905C34163AEF2246FD69016CE24A7BA40F8E7E42EDAC2D0A44605
Hash of sig:    F8383917AF79C8E540D2356AB05F08B465BF32DFEC444B787CE31BF48CC6C3DD

Test case 87-LN-09
Seed:                   21212285BED53B3411705DAF5F3BDDB6F0618EB571B36EE11A74053407A269F5
Hash of Keys:   737061155A9A03F11F9FEBBB940BED4DD54542C4A6212F89A5EB4EC2BE542782
Message:                FFE38246BF3DEFD9CAD15CC17CEA511C067D582E04227B479E32F9197CF91482
Hash of sig:    C4C12C58032052FB2D21F0C6A7388A63154FB85B74287D2859DE6C1C6F7F277B

Test case 87-LN-10
Seed:                   A2744470587C71BA43EC26DC390CE3531978F315993C653E5D3EFD2849D5D9F1
Hash of Keys:   B1BF37BFFB11531B6ADD697870D7DB2E2462D0A97A63F09C1D0038457C6D795A
Message:                9831A830231A160B9847203341A5F30BF3E87A2A482AEEA6886315C92B5C4E4C
Hash of sig:    46C669D2FEB643A38E54FF87B790CC33F44043A1B6B31DB9474D301328CA2A7F
```

FCS_COP.1/SigVer Cryptographic Operation - Signature Verification

FCS_COP.1.1/SigVer

The TSF shall perform [*digital signature verification*] in accordance with a specified cryptographic algorithm [**selection**: *Cryptographic Algorithm*] and cryptographic key sizes [**selection**: *Cryptographic Key Sizes*] that meet the following: [**selection**: *List of Standards*]

The following table provides the allowable choices for completion of the selection operations in FCS_COP.1/SigVer.

Table 16: Allowable choices for FCS_COP.1/SigVer

Table 10: Allowable choices for FCS_COP.1/SigVer

Identifier	Cryptographic Algorithm	Cryptographic Key Sizes	List of Standards
RSA-PKCS	RSASSA-PKCS1-v1_5	Modulus of size [**selection**: *3072*, *4096*, *6144*, *8192*] bits and hash [**selection**: *SHA-384*, *SHA-512*]	RFC 8017 (Section 8.2) [PKCS #1 v2.2] FIPS PUB 186-5 (Section 5.4) [RSASSA-PKCS1-v1_5]
RSA-PSS	RSASSA-PSS	Modulus of size [**selection**: *3072*, *4096*, *6144*, *8192*] bits and hash [**selection**: *SHA-384*, *SHA-512*]	RFC 8017 (Section 8.1) [PKCS#1 v2.2] FIPS PUB 186-5 (Section 5.4) [RSASSA-PSS]
ECDSA	ECDSA	Elliptic Curve [**selection**: *P-384*, *P-521*] using hash [**selection**: *SHA-384*, *SHA-512*]	[**selection**: *ISO/IEC 14888-3:2018 (Subclause 6.6)*, *FIPS PUB 186-5 (Section 6.4.2)*] [ECDSA] NIST SP-800 186 (Section 4) [NIST Curves]
LMS	LMS	Private key size =undefined Winternitz parameter =undefined Tree height = [**selection**: *5*, *10*, *15*, *20*, *25*]	RFC 8554 [LMS] NIST SP 800-208 [parameters]
XMSS	XMSS	Private key size =undefined Tree height = [**selection**: *10*, *16*, *20*]	RFC 8391 [XMSS] NIST SP 800-208 [parameters]
ML-DSA	ML-DSA Signature Verification	Parameter set = ML-DSA-87	NIST FIPS 204 (Section 5.3)

Application Note: The ST author should choose the algorithm implemented to perform digital signatures; if more than one algorithm is available, this requirement should be iterated to specify the functionality. For the algorithm chosen, the ST author should make the appropriate assignments and selections to specify the parameters that are implemented for that algorithm.

Evaluation Activities

FCS_COP.1/SigVer
TSS
The evaluator shall examine the TSS to verify that any one-time values such as nonces or masks are constructed and used in accordance with the relevant standards.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
The following tests are conditional based on the selections made in the SFR. The evaluator shall perform the following tests or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

RSA-PKCS Signature Verification

Identifier	*Cryptographic Algorithm Parameters*	*Cryptographic Key Sizes*	*List of Standards*
RSA-PKCS	*RSASSA-PKCS1-v1_5*	*Modulus of size [**selection:** 3072, 4096, 6144, 8192] bits, hash [**selection:** SHA-384, SHA-512]*	*RFC 8017 (Section 8.2) [PKCS #1 v2.2]* *NIST FIPS PUB 186-5 (Section 5.4) [RSASSA-*

			PKCS1-v1_5]

To test the TOE's ability to perform RSA Digital Signature Verification using PKCS1-v1,5 signature type, the evaluator shall perform Generated Data Test using the following input parameters:

- *Modulus size [3072, 4096, 6144, 8192] bits*
- *Hash algorithm [SHA-384, SHA-512]*

Generated Data Test

For each supported combination of the above parameters, the evaluator shall cause the TOE to generate six test cases using a random message and its signature such that the test cases are modified as follows:

- *One test case is left unmodified*
- *For one test case the Message is modified*
- *For one test case the Signature is modified*
- *For one test case the exponent (e) is modified*
- *For one test case the IR is moved*
- *For one test case the Trailer is moved*

The TOE must correctly verify the unmodified signatures and fail to verify the modified signatures.

RSA-PSS Signature Verification

Identifier	***Cryptographic Algorithm Parameters***	***Cryptographic Key Sizes***	***List of Standards***
RSA-PSS	*RSASSA-PSS*	*Modulus of size [**selection:** 3072, 4096, 6144, 8192] bits, hash [**selection:** SHA-384, SHA-512]*	*RFC 8017 (Section 8.2) [PKCS #1 v2.2]* *NIST FIPS PUB 186-5 (Section 5.4) [RSASSA-PSS]*

To test the TOE's ability to perform RSA Digital Signature Verification using PSS signature type, the evaluator shall perform the Generated Data Test using the following input parameters:

- *Modulus size [3072, 4096, 6144, 8192] bits*
- *Hash algorithm [SHA-384, SHA-512]*
- *Salt length [0-hash length]*
- *Mask function [MGF1]*

Generated Data Test

For each supported combination of the above parameters, the evaluator shall cause the TOE to generate six test cases using random data such that the test cases are modified as follows:

- *One test case is left unmodified*
- *For one test case the Message is modified*
- *For one test case the Signature is modified*
- *For one test case the exponent (e) is modified*
- *For one test case the IR is moved*
- *For one test case the Trailer is moved*

The TOE must correctly verify the unmodified signatures and fail to verify the modified signatures.

ECDSA Signature Verification

Identifier	***Cryptographic Algorithm Parameters***	***Cryptographic Key Sizes***	***List of Standards***
ECDSA	*ECDSA*	*Elliptic Curve [**selection:** P-384, P-521] and hash function using [**selection:** SHA-384, SHA-512]*	*[**selection:** ISO/IEC 14888-3:2018 (Subclause 6.6), NIST FIPS PUB 186-5 (Sections 6.3.1, 6.4.1] [ECDSA]* *NIST SP-800 186 (Section 4) [NIST Curves]*

To test the TOE's ability to perform ECDSA Digital Signature Verification, the evaluator shall perform the Algorithm Functional Test using the following input parameters:

- *Elliptic Curve [P-384, P-521]*
- *Hash algorithm [SHA-384, SHA-512]*

Algorithm Functional Test

For each supported combination of the above parameters, the evaluator shall cause the TOE to

generate test cases consisting of messages and signatures such that the 21 test cases are modified as follows:

- *Three test cases are left unmodified*
- *For three test cases the Message is modified*
- *For three test cases the key is modified*
- *For three test cases the r value is modified*
- *For three test cases the s value is modified*
- *For three test cases the value r is zeroed*
- *For three test cases the value s is zeroed*

The TOE must correctly verify the unmodified signatures and fail to verify the modified signatures.

LMS Signature Verification

Identifier	***Cryptographic Algorithm Parameters***	***Cryptographic Key Sizes***	***List of Standards***
LMS	*LMS*	*Private key size = [**selection:** 192 bits with [**selection:** SHA256/192, SHAKE256/192], 256 bits with [**selection:** SHA-256, SHAKE256]], Winternitz parameter = [**selection:** 1, 2, 4, 8], and tree height = [**selection:** 5, 10, 15, 20, 25]*	*RFC 8554 [LMS]* *NIST SP 800-208 [parameters]*

To test the TOE's ability to verify cryptographic digital signature using LMS, the evaluator shall perform the Algorithm Functional Test using the following input parameters:

- *Hash algorithm [SHA-256/192, SHAKE256/192, SHA-256, SHAKE256]*
- *Winternitz [1, 2, 4, 8]*
- *Tree height [5, 10, 15, 20, 25]*

Algorithm Functional Test

For each supported combination of the above parameters, the evaluator shall generate four test cases consisting of signed messages and keys, such that

- *One test case is unmodified (i.e. correct)*
- *For one test case modify the message, i.e. the message is different*
- *For one test case modify the signature, i.e. signature is different*
- *For one test case modify the signature header so that it is a valid header for a different LMS parameter set.*

The TOE must correctly verify the unmodified test case and fail to verify the modified test cases.

XMSS Signature Verification

Identifier	***Cryptographic Algorithm Parameters***	***Cryptographic Key Sizes***	***List of Standards***
XMSS	*XMSS*	*Private key size = [**selection:** 192 bits with [**selection:** SHA256/192, SHAKE256/192], 256 bits with [**selection:** SHA-256, SHAKE256]], and tree height = [**selection:** 10, 16, 20]*	*RFC 8391 [XMSS]* *NIST SP 800-208 [parameters]*

To test the TOE's ability to verify digital signatures using XMSS or XMSS MT, the evaluator shall perform the XMSS digital signature verification test using the following input parameters:

- *Hash algorithm [SHA-256/192, SHAKE256/192, SHA-256, SHAKE256]*
- *Tree height [10, 16, 20]*

XMSS Digital Signature Verification Test

For each supported combination of the above parameters, the evaluator shall generate four test cases consisting of signed messages and keys, such that

- *One test case is unmodified (i.e. correct)*
- *For one test case modify the message, i.e. the message is different*
- *For one test case modify the signature, i.e. signature is different*
- *For one test case modify the signature header so that it is a valid header for a different XMSS parameter set*

The evaluator shall verify the correctness of the implementation by verifying that the TOE correctly verifies the unmodified test case and fails to verify the modified test cases.

ML-DSA Signature Verification

Identifier	***Cryptographic Algorithm Parameters***	***Cryptographic Key Sizes***	***List of Standards***

ML-DSA	*ML-DSA SigVer*	*Parameter set = ML-DSA-87*	*NIST FIPS PUB 204 (Section 5.2)*

To test the TOE's ability to validate digital signatures using ML-DSA, the evaluator shall perform the Algorithm Functional Test using the following input parameters:

- *Parameter set [ML-DSA-87]*
- *Previously generated signed Message [8-65535] bytes*
- *Mu value (if generated externally)*
- *Context (for external interface testing)*
- *Previously generated public key (pk)*
- *Previously generated Signature*

Algorithm Functional Test

For each combination of supported parameter set and capabilities, the evaluator shall require the implementation under test to validate 15 signatures. Each group of 15 test cases is modified as follows:

- *Three test cases are left unmodified*
- *For three test cases the Signed message is modified*
- *For three test cases the component of the signature that commits the signer to the message is modified*
- *For three test cases the component of the signature that allows the verifier to construct the vector z is modified*
- *For three test cases the component of the signature that allows the verifier to construct the hint array is modified*

The TOE must correctly verify the unmodified signatures and fail to verify the modified signatures.

FCS_COP.1/SKC Cryptographic Operation - Encryption/Decryption

This is a selection-based component. Its inclusion depends upon selection from FDP_DAR_EXT.1.2.

FCS_COP.1.1/SKC

The TSF shall perform [*symmetric-key encryption and decryption*] in accordance with a specified cryptographic algorithm [**selection**: *Cryptographic Algorithm*] and cryptographic key sizes [**selection**: *Cryptographic Key Sizes*] that meet the following: [**selection**: *List of Standards*]

The following table provides the allowable choices for completion of the selection operations in FCS_COP.1/SKC.

Table 17: Allowable choices for FCS_COP.1/SKC

Identifier	Cryptographic Algorithm	Cryptographic Key Sizes	List of Standards
AES-CBC	AES in CBC mode with non-repeating and unpredictable IVs	256 bits	[**selection**: *ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197*] [AES] [**selection**: *ISO/IEC 10116:2017 (Clause 7), NIST SP 800-38A*] [CBC]
XTS-AES	AES in XTS mode with unique tweak values that are consecutive non-negative integers starting at an arbitrary non-negative integer	512 bits	[**selection**: *ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197*] [AES] [**selection**: *IEEE Std. 1619-2018, NIST SP 800-38E*] [XTS]

Evaluation Activities

FCS_COP.1/SKC

TSS

The evaluator shall examine the TSS to ensure that it describes the construction of any IVs, tweak values, and counters in conformance with the relevant specifications.

If XTS-AES is claimed then the evaluator shall examine the TSS to verify that the TOE creates full-length keys by methods that ensure that the two key halves are different and independent.

Guidance

There are no additional Guidance evaluation activities for this component.

Tests

The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products.

The following tests are conditional based on the selections made in the SFR. The evaluator shall perform the following tests or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

AES-CBC

Identifier	***Cryptographic Algorithm***	***Cryptographic Key Sizes***	***List of Standards***
AES-CBC	*AES in CBC mode with non-repeating and unpredictable IVs*	*256 bits*	*[**selection:** ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES]* *[**selection:** ISO/IEC 10116:2017 (Clause 7), NIST SP 800-38A] [CBC]*

To test the TOE's ability to encrypt and decrypt data using AES in CBC mode, the evaluator shall perform Algorithm Functional Tests and Monte Carlo Tests using the following input parameters:

- *Key size [256] bits*
- *Direction [encryption, decryption]*

Algorithm Functional Tests

Algorithm Functional Tests are designed to verify the correct operation of the logical components of the algorithm implementation under normal operation using different block sizes. For AES-CBC, there are two types of AFTs:

Known-Answer Tests

For each combination of direction and claimed key size, the TOE must be tested using the GFSBox, KeySbox, VarTxt, and VarKey test cases listed in Appendixes B through E of The Advanced Encryption Standard Algorithm Validation Suite (AESAVS), NIST, 15 November 2002.

Multi-Block Message Tests

For each combination of direction and claimed key size, the TOE must be tested against 10 test cases consisting of a random IV, random key, and random plaintext or ciphertext. The plaintext or ciphertext starts with a length of 16 bytes and increases by 16 bytes for each test case until reaching 160 bytes.

Monte Carlo Tests

Monte Carlo tests are intended to test the implementation under strenuous conditions. The TOE must process the test cases according to the following algorithm once for each combination of direction and key size:

```
Key[0] = Key
IV[0] = IV
PT[0] = PT
for i = 0 to 99 {
Output Key[i], IV[i], PT[0]
for j = 0 to 999 {
if (j == 0) {
CT[j] = AES-CBC-Encrypt(Key[i], IV[i], PT[j])
PT[j+1] = IV[i]
} else {
CT[j] = AES-CBC-Encrypt(Key[i], PT[j])
PT[j+1] = CT[j-1]
}
}
Output CT[j]
AES_KEY_SHUFFLE(Key, CT)
```

```
IV[i+1] = CT[j]
PT[0] = CT[j-1]
}
```

where

AES_KEY_SHUFFLE

is defined as:

```
If ( keylen = 128 )
Key[i+1] = Key[i] xor MSB(CT[j], 128)
If ( keylen = 192 )
Key[i+1] = Key[i] xor (LSB(CT[j-1], 64) || MSB(CT[j], 128))
If ( keylen = 256 )
Key[i+1] = Key[i] xor (MSB(CT[j-1], 128) || MSB(CT[j], 128))
```

The above pseudocode is for encryption. For decryption, swap all instances of CT and PT.

The initial IV, key, and plaintext or ciphertext should be random.

The evaluator shall test the decrypt functionality using the same test as above, exchanging CT and PT, and replacing AES-CBC-Encrypt with AES-CBC-Decrypt.

XTS-AES

Identifier	***Cryptographic Algorithm***	***Cryptographic Key Sizes***	***List of Standards***
XTS-AES	*AES in XTS mode with unique tweak values that are consecutive non-negative integers starting at an arbitrary non-negative integer*	*512 bits*	***[selection:*** *ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES]* ***[selection:*** *IEEE Std. 1619-2018, NIST SP 800-38E] [XTS]*

To test the TOE's ability to encrypt and decrypt data using AES in XTS mode, the evaluator shall perform the Single Data Unit Test and the Multiple Data Unit Test using the following input parameters:

- *Direction [encryption, decryption]*
- *Key size [512] bits*
- *Tweak value format [128-bit hex string, data unit sequence number]*

Single Data Unit Test

For each combination of claimed key size, direction, and supported tweak value format, the evaluator shall generate 50 test cases consisting of random payload data. The payload data size is determined randomly for each test case from supported values within the range [128-65536] bits. The payload size and data unit size must be equal.

Multiple Data Unit Test

For each combination of claimed key size, direction, and supported tweak value format, the evaluator shall generate 50 test cases consisting of random payload data. The payload data size is determined randomly for each test case from supported values within the range [128-65536] bits. Likewise, the data unit size is determined randomly for each test case from supported values within the range [128-65535] bits. The payload size and data unit size must not be equal.

The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated by a known good implementation using the same input parameters.

FCS_COP.1/XOF Cryptographic Operation - Extendable-Output Function

This is a selection-based component. Its inclusion depends upon selection from FCS_CKM.1.1/AKG, FCS_COP.1.1/SigVer.

FCS_COP.1.1/XOF

The TSF shall perform [*extendable-output function*] in accordance with a specified cryptographic algorithm [**selection**: *Cryptographic Algorithm*] and parameters [**selection**: *Parameters*] that meet the following: [**selection**: *List of Standards*]

The following table provides the allowable choices for completion of the selection operations of FCS_COP.1/XOF.

Table 18: Allowable choices for FCS_COP.1/XOF

Cryptographic Algorithm	Parameters	List of Standards
SHAKE	Functions = [SHAKE128, SHAKE256]	NIST FIPS PUB 202 Section 6.2 [SHAKE]

Application Note: In accordance with CNSA 2.0, SHAKE is permitted to be used only as a component of LMS or XMSS. Therefore this component is claimed only if LMS or XMSS is claimed in FCS_COP.1/SigVer.

Since LMS and XMSS use both SHAKE128 and SHAKE256 internally, claiming and testing of both functions is mandatory.

Evaluation Activities

FCS_COP.1/XOF
TSS
There are no additional TSS evaluation activities for this component.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
The following tests are conditional based on the selections made in the SFR. The evaluator shall perform the following tests or witness respective tests executed by the developer. The tests must be executed on a platform that is as close as practically possible to the operational platform (but which may be instrumented in terms of, for example, use of a debug mode). Where the test is not carried out on the TOE itself, the test platform shall be identified and the differences between test environment and TOE execution environment shall be described.

SHAKE

Cryptographic Algorithm	*Parameters*	*List of Standards*
SHAKE	*Function = [SHAKE128, SHAKE256]*	*NIST FIPS PUB 202 Section 6.2 [SHAKE]*

To test the TOE's implementation of the SHAKE Extendable Output Function the evaluator shall perform the Algorithm Functional Test, Monte Carlo Test, and Variable Output Test using the following input parameters:

- *Function [SHAKE128, SHAKE256]*
- *Output length [16-65536] bits*

Algorithm Functional Test

For each supported function, generate test cases consisting of random data for every message length from 0 bits (if supported) to rate-1 bits, where rateequals

- *1344 for SHAKE128, and*
- *1088 for SHAKE256.*

Additionally, generate tests cases of random data for messages of every multiple of (rate+1) bits starting at length rate, and continuing until 65535 is exceeded. For SHAKE128, this should result in a total of 1391 test cases.

Monte Carlo Test

The Monte Carlos test takes in a single 128-bit message (SEED) and desired output length in bits, and runs 100 iterations of the chained computation. MaxOutBytes and MinOutBytes are the largest and smallest supported input and output sizes in bytes, respectively.

```
Range = maxOutBytes - minOutBytes + 1
OutputLen = maxOutBytes
For j = 0 to 99
MD[0] = SEED
For i = 1 to 1000
MSG[i] = 128 leftmost bits of MD[i-1]
if (MSG[i] < 128 bits)
Append 0 bits on rightmost side of MSG[i] til MSG[i] is 128 bits
MD[i] = SHAKE(MSG[i], OutputLen * 8)

RightmostOutputBits = 16 rightmost bits of MD[i] as an integer
OutputLen = minOutBytes + (RightmostOutputBits % Range)
Output MD[1000], OutputLen
SEED = MD[1000]
```

Variable Output Test

This test measures the ability of the TOE to generate output digests of varying sizes.

The evaluator shall generate 512 test cases such that the input for each test case consists of 128- bits of random data, and the output length includes the minimum supported value, the maximum supported value, and 510 random values between the minimum and maximum digest

sizes supported by the implementation.

FCS_HTTPS_EXT.1 HTTPS Protocol

FCS_HTTPS_EXT.1.1

The TSF shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2

The TSF shall implement HTTPS using TLS as defined in [*the Functional Package for Transport Layer Security (TLS), version 2.1*].

Application Note: The Functional Package for Transport Layer Security (TLS), version 2.1 must be included in the ST, with the following selections made:

- FCS_TLS_EXT.1:
  - TLS must be selected
  - Client must be selected

FCS_HTTPS_EXT.1.3

The TSF shall notify the application and [**selection**: *not establish the connection, request application authorization to establish the connection, no other action*] if the peer certificate is deemed invalid.

Application Note: Validity is determined by the certificate path, the expiration date, and the revocation status in accordance with RFC 5280.

The selection made for FCS_HTTPS_EXT.1.3 must be consistent with the claims made in FCS_TLSC_EXT.1.6 of Functional Package for Transport Layer Security (TLS), version 2.1, e.g. if the TLS rejects all invalid certificates with no exceptions, then "not establish the connection" will be selected here. FMT_SMF_EXT.1 Function 23 configures whether to allow or disallow the establishment of a trusted channel if the peer certificate is deemed invalid.

Evaluation Activities

FCS_HTTPS_EXT.1
TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests

- *Test FCS_HTTPS_EXT.1:1: The evaluator shall attempt to establish an HTTPS connection with a webserver, observe the traffic with a packet analyzer, and verify that the connection succeeds and that the traffic is identified as TLS or HTTPS.*

  Other tests are performed in conjunction with testing in the Functional Package for Transport Layer Security (TLS), version 2.1.

  Certificate validity shall be tested in accordance with testing performed for FIA_X509_EXT.1 as defined in Functional Package for X.509, version 1.0, and the evaluator shall perform the following test:
- *Test FCS_HTTPS_EXT.1:2: The evaluator shall demonstrate that using a certificate without a valid certification path results in an application notification. Using the administrative guidance, the evaluator shall then load a certificate or certificates to the Trust Anchor Database needed to validate the certificate to be used in the function, and demonstrate that the function succeeds. The evaluator then shall delete one of the certificates, and show that the application is notified of the validation failure.*

FCS_IV_EXT.1 Initialization Vector Generation

FCS_IV_EXT.1.1

The TSF shall generate IVs in accordance with [*: References and IV Requirements for NIST-approved Cipher Modes*].

Application Note: lists the requirements for composition of IVs according to the NIST Special Publications for each cipher mode. The composition of IVs generated for encryption according to a cryptographic protocol is addressed by the protocol. Thus, this requirement addresses only IVs generated for key storage and data storage encryption.

Evaluation Activities

FCS_IV_EXT.1
TSS
The evaluator shall examine the key hierarchy section of the TSS to ensure that the encryption of all keys is described and the formation of the IVs for each key encrypted by the same KEK meets FCS_IV_EXT.1.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FCS_RBG.1 Random Bit Generation (RBG)

FCS_RBG.1.1

The TSF shall perform deterministic random bit generation services using [**selection**:

- *Hash_DRBG (any)*
- *HMAC_DRBG (any)*
- *CTR_DRBG (AES)*

] in accordance with [*NIST SP 800-90A*] after initialization with a seed.

Application Note: NIST SP 800-90A contains three different methods of generating random numbers; each of these, in turn, depends on underlying cryptographic primitives (hash functions/ciphers). The ST author will select the function used and include the specific underlying cryptographic primitives used in the requirement or in the TSS. Only the hash function SHA-384 is allowed for Hash_DRBG or HMAC_DRBG, only AES-based implementations for CTR_DRBG are allowed.

FCS_RBG.1.2

The TSF shall use a [**selection**: *TSF noise source [**assignment**: name of noise source], **multiple TSF noise sources [assignment: names of noise sources]** , TSF interface for seeding*] for initialized seeding.

Application Note: For the selection in this requirement, the ST author selects "TSF noise source" if a single noise source is used as input to the DRBG. The ST author selects "multiple TSF noise sources" if a seed is formed from a combination of two or more noise sources within the TOE boundary. If the TSF implements two or more separate DRBGs that are seeded in separate manners, this SFR should be iterated for each DRBG. It multiple distinct noise sources exist such that each DRBG only uses one of them, then each iteration would select "TSF noise source"; "multiple TSF noise sources" is only selected if a single DRBG uses multiple noise sources for its seed. The ST author selects "TSF interface for seeding" if noise source data is generated outside the TOE boundary.
If "TSF noise source" is selected, FCS_RBG.3 must be claimed.

If "multiple TSF noise sources" is selected, FCS_RBG.4 and FCS_RBG.5 must be claimed.

If "TSF interface for seeding" is selected, FCS_RBG.2 must be claimed.

FCS_RBG.1.3

The TSF shall update the RBG state by [**selection**: *reseeding, uninstantiating and reinstantiating*] using a [**selection**: *TSF noise source [**assignment**: name of noise source], TSF interface for seeding*] in the following situations: [**selection**:

- *never*
- *on demand*
- *on the condition: [**assignment**: condition]*
- *after [**assignment**: time]*

] in accordance with [**assignment**: *list of standards*].

Evaluation Activities

FCS_RBG.1.1
TSS
The evaluator shall verify that the TSS identifies the DRBGs used by the TOE.

Guidance
If the DRBG functionality is configurable, the evaluator shall verify that the operational guidance includes instructions on how to configure this behavior.

Tests

The evaluator shall perform the following tests:

The evaluator shall perform 15 trials for the RNG implementation. If the RNG is configurable, the evaluator shall perform 15 trials for each configuration. The evaluator shall also confirm that the operational guidance contains appropriate instructions for configuring the RNG functionality.

If the RNG has prediction resistance enabled, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) generate a second block of random bits (4) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 – 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. "generate one block of random bits" means to generate random bits with number of returned bits equal to the Output Block Length (as defined in NIST SP 800-90A).

If the RNG does not have prediction resistance, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) reseed, (4) generate a second block of random bits (5) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 – 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The fifth value is additional input to the first call to generate. The sixth and seventh are additional input and entropy input to the call to reseed. The final value is additional input to the second generate call.

The following list contains more information on some of the input values to be generated/selected by the evaluator.

- ***Entropy input:** The length of the entropy input value must equal the seed length.*
- ***Nonce:** If a nonce is supported (CTR_DRBG with no Derivation Function does not use a nonce), the nonce bit length is one-half the seed length.*
- ***Personalization string:** The length of the personalization string must be less than or equal to seed length. If the implementation only supports one personalization string length, then the same length can be used for both values. If more than one string length is support, the evaluator shall use personalization strings of two different lengths. If the implementation does not use a personalization string, no value needs to be supplied.*
- ***Additional input:** The additional input bit lengths have the same defaults and restrictions as the personalization string lengths.*

FCS_RBG.1.2
Documentation will be produced - and the evaluator shall perform the activities - in accordance with Appendix D - Entropy Documentation And Assessment and the Clarification to the Entropy Documentation and Assessment Annex.

FCS_RBG.1.3
TSS
The evaluator shall verify that the TSS identifies how the DRBG state is updated, and the situations under which this may occur.

Guidance
If the ST claims that the DRBG state can be updated on demand, the evaluator shall verify that the operational guidance has instructions for how to perform this operation.

Tests
There are no test activities for this element.

FCS_RBG.2 Random Bit Generation (External Seeding)

This is a selection-based component. Its inclusion depends upon selection from FCS_RBG.1.2.

FCS_RBG.2.1

The TSF shall be able to accept a minimum input of [**assignment**: *minimum input length greater than zero*] from a TSF interface for the purpose of seeding.

Application Note: This requirement is claimed when a DRBG is seeded with entropy from one or more noise source that is outside the TOE boundary. In the case of a mobile device this would likely only occur when the entropy source is a Dedicated Security Component that is integrated with the TOE. Typically the entropy produced by an environmental noise source is conditioned such that the input length has full entropy and is therefore usable as the seed. However, if this is not the case, it should be noted what the minimum entropy rate of the noise source is so that the TSF can collect a sufficiently large sample of noise data to be conditioned into a seed value.

Evaluation Activities

FCS_RBG.2
The evaluator shall examine the entropy documentation required by FCS_RBG.1.2 to verify that

it identifies, for each DRBG function implemented by the TOE, the TSF external interface used to seed the TOE's DRBG. The evaluator shall verify that this includes the amount of sampled data and the min-entropy rate of the sampled data such that it can be determined that sufficient entropy can be made available for the highest strength keys that the TSF can generate (e.g., 256 bits). If the seed data cannot be assumed to have full entropy (e.g., the min-entropy of the sampled bits is less than 1), the evaluator shall ensure that the entropy documentation describes the method by which the TOE estimates the amount of entropy that has been accumulated to ensure that sufficient data is collected and any conditioning that the TSF applies to the output data to create a seed of sufficient size with full entropy.

TSS
There are no additional TSS evaluation activities for this component.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
There are no test activities for this component.

FCS_RBG.3 Random Bit Generation (Internal Seeding - Single Source)

This is a selection-based component. Its inclusion depends upon selection from FCS_RBG.1.2.

FCS_RBG.3.1

The TSF shall be able to seed the RBG using a [**selection, choose one of**: *TSF software-based noise source*, *TSF hardware-based noise source [**assignment**: name of noise source]*] with a minimum of [**assignment**: *number of bits*] bits of min-entropy.

Application Note: This requirement is claimed when a DRBG is seeded with entropy from a single noise source that is within the TOE boundary. Min-entropy should be expressed as a ratio of entropy bits to sampled bits so that the total amount of data needed to ensure full entropy is known, as well as the conditioning function by which that data is reduced in size to the seed.

Evaluation Activities

FCS_RBG.3
The evaluator shall examine the entropy documentation required by FCS_RBG.1.2 to verify that it identifies, for each DRBG function implemented by the TOE, the TSF noise source used to seed the TOE's DRBG. The evaluator shall verify that this includes the amount of sampled data and the min-entropy rate of the sampled data such that it can be determined that sufficient entropy can be made available for the highest strength keys that the TSF can generate (e.g., 256 bits). If the seed data cannot be assumed to have full entropy (e.g., the min-entropy of the sampled bits is less than 1), the evaluator shall ensure that the entropy documentation describes the method by which the TOE estimates the amount of entropy that has been accumulated to ensure that sufficient data is collected and any conditioning that the TSF applies to the output data to create a seed of sufficient size with full entropy.

TSS
There are no additional TSS evaluation activities for this component.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
There are no test activities for this component.

FCS_RBG.4 Random Bit Generation (Internal Seeding - Multiple Sources)

This is a selection-based component. Its inclusion depends upon selection from FCS_RBG.1.2.

FCS_RBG.4.1

The TSF shall be able to seed the RBG using [**selection**: *[**assignment**: number] TSF software-based noise source(s), [**assignment**: number] TSF hardware-based noise source(s)*].

Application Note: This requirement is claimed when a DRBG is seeded with entropy from multiple noise sources that are within the TOE boundary. FCS_RBG.5 defines the mechanism by which these sources are combined to ensure sufficient minimum entropy.

Evaluation Activities

FCS_RBG.4
The evaluator shall examine the entropy documentation required by FCS_RBG.1.2 to verify that it identifies, for each DRBG function implemented by the TOE, each TSF noise source used to seed the TOE's DRBG. The evaluator shall verify that this includes the amount of sampled data and the min-entropy rate of the sampled data from each data source.

TSS
There are no additional TSS evaluation activities for this component.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
There are no test activities for this component.

FCS_RBG.5 Random Bit Generation (Combining Noise Sources)

This is a selection-based component. Its inclusion depends upon selection from FCS_RBG.1.2.

FCS_RBG.5.1

The TSF shall [**assignment**: *combining operation*] [**selection**: *output from TSF noise source(s), input from TSF interface(s) for seeding*] to create the entropy input into the derivation function as defined in [**assignment**: *list of standards*], resulting in a minimum of [**assignment**: *number of bits*] bits of min-entropy.

Evaluation Activities

FCS_RBG.5
Using the entropy sources specified in FCS_RBG.4, the evaluator shall examine the entropy documentation required by FCS_RBG.1.2 to verify that it describes the method by which the various entropy sources are combined into a single seed. This should include an estimation of the rate at which each noise source outputs data and whether this is dependent on any system-specific factors so that each source's relative contribution to the overall entropy is understood. The evaluator shall verify that the resulting combination of sampled data and the min-entropy rate of the sampled data is described in sufficient detail to determine that sufficient entropy can be made available for the highest strength keys that the TSF can generate (e.g., 256 bits). If the seed data cannot be assumed to have full entropy (e.g., the min-entropy of the sampled bits is less than 1), the evaluator shall ensure that the entropy documentation describes the method by which the TOE estimates the amount of entropy that has been accumulated to ensure that sufficient data is collected and any conditioning that the TSF applies to the output data to create a seed of sufficient size with full entropy.

TSS
There are no additional TSS evaluation activities for this component.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
There are no test activities for this component.

FCS_RBG.6 Random Bit Generation Service

FCS_RBG.6.1

The TSF shall provide a [**selection**: *hardware*, *software*, *[**assignment**: other interface type]*] interface to make the RBG output, as specified in FCS_RBG.1 Random bit generation (RBG), available as a service to entities outside of the TOE.

Evaluation Activities

FCS_RBG.6
TSS
The evaluator shall verify that the TSS identifies the interface that the TSF makes available for calling applications to obtain DRBG output.

Guidance
The evaluator shall verify that the guidance documentation includes an API specification for the random bit generation service such that it is clear how a calling application is able to obtain DRBG output from the TSF.

Tests
The evaluator shall invoke the API specified by the guidance documentation to determine that the TSF provides DRBG output upon proper invocation of the API.

FCS_RBG_EXT.2 Random Bit Generator State Preservation

This is an objective component.

FCS_RBG_EXT.2.1

The TSF shall save the state of the deterministic RBG at power-off, and shall use this state as input to the deterministic RBG at startup.

Application Note: The capability to add the state saved at power-off as input to the RBG prevents an RBG that is slow to gather entropy from producing the same output regularly and across reboots. Since there is no guarantee of the protections provided when the state is stored (or a requirement for any such protection), it is assumed that the state is 'known', and therefore cannot contribute entropy to the RBG, but can introduce enough variation that the initial RBG values are not predictable and exploitable.

Evaluation Activities

FCS_RBG_EXT.2
TSS
The evaluation activity for this requirement is captured in the RBG documentation for . The evaluator shall verify that the documentation describes how the state is generated so as to be available for the next startup, how the state is used as input to the DRBG, and any protection measures used for the state while the TOE is powered off.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FCS_RBG_EXT.3 Support for Personalization String

This is an objective component.

FCS_RBG_EXT.3.1

The TSF shall allow applications to add data to the deterministic RBG using the Personalization String as defined in SP 800-90A.

Application Note: As specified in SP 800-90A, the TSF must not count data input from an application towards the entropy required by FCS_RBG.1. Thus, the TSF must not allow the only input to the RBG seed to be from an application.

Evaluation Activities

FCS_RBG_EXT.3
The evaluator shall verify that this function is included as an interface to the RBG in the documentation required by and that the behavior of the RBG following a call to this interface is described. The evaluator shall also verify that the documentation of the RBG describes the conditions of use and possible values for the Personalization String input to the SP 800-90A specified DRBG.

TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests
- *Test FCS_RBG_EXT.3:1: The evaluator shall write, or the developer shall provide, an application that adds data to the RBG via the Personalization String. The evaluator shall verify that the request succeeds.*

FCS_SRV_EXT.1 Cryptographic Algorithm Services

FCS_SRV_EXT.1.1

The TSF shall provide a mechanism for applications to request the TSF to

perform the following cryptographic operations: [

- *All mandatory and [**selection**: selected algorithms, selected algorithms with the exception of ECC over curve 25519-based algorithms] in FCS_CKM_EXT.7/LOCKED*
- *The following algorithms in FCS_COP.1/ENCRYPT: AES-CBC, [**selection**: AES Key Wrap, AES Key Wrap with Padding, AES-GCM, AES-CCM, no other modes]*
- *All selected algorithms in FCS_COP.1/SIGN*
- *All mandatory and selected algorithms in FCS_COP.1/Hash*
- *All mandatory and selected algorithms in FCS_COP.1/KeyedHash*
- *[**selection**:*
  - *All mandatory and [**selection**: selected algorithms, selected algorithms with the exception of ECC over curve 25519-based algorithms] in FCS_CKM.1/AKG or FCS_CKM.1/SKG*
  - *The selected algorithms in FCS_CKM_EXT.8*
  - *No other cryptographic operations*

  ]

].

Application Note: For each of the listed FCS components in the bulleted list, the intent is that the TOE will make available all algorithms specified for that component in the ST.

The exception is for FCS_COP.1/ENCRYPT. The TOE is not required to make available AES_CCMP, AES_XTS, AES_GCMP-256, or AES_CCMP_256 even though they may be implemented to perform TSF-related functions. It is acceptable for the platform to not provide AES Key Wrap (KW) and AES Key Wrap with Padding (KWP) to applications even if selected in FCS_COP.1/ENCRYPT. However, the ST author is expected to select AES-GCM or AES-CCM if it is selected in the ST for the FCS_COP.1/ENCRYPT component.

Evaluation Activities

FCS_SRV_EXT.1
The evaluator shall verify that the API documentation provided according to Section 5.2.2 Class ADV: Development includes the security functions (cryptographic algorithms) described in these requirements.

TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests
The evaluator shall write, or the developer shall provide access to, an application that requests cryptographic operations by the TSF. The evaluator shall verify that the results from the operation match the expected results according to the API documentation. This application may be used to assist in verifying the cryptographic operation Evaluation Activities for the other algorithm services requirements.

FCS_SRV_EXT.2 Cryptographic Key Storage Services

This is an objective component.

FCS_SRV_EXT.2.1

The TSF shall provide a mechanism for applications to request the TSF to perform the following cryptographic operations: [

- *Algorithms in FCS_COP.1/ENCRYPT*
- *Algorithms in FCS_COP.1/SIGN*

] **by keys stored in the secure key storage.**

Application Note: The TOE will, therefore, be required to perform cryptographic operations on behalf of applications using the keys stored in the TOE's secure key storage.

Evaluation Activities

FCS_SRV_EXT.2
The evaluator shall verify that the API documentation for the secure key storage includes the cryptographic operations by the stored keys.

TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests
The evaluator shall write, or the developer shall provide access to, an application that requests cryptographic operations of stored keys by the TSF. The evaluator shall verify that the results from the operation match the expected results according to the API documentation. The evaluator shall use these APIs to test the functionality of the secure key storage according to the Evaluation Activities in FCS_STG_EXT.1.

FCS_STG_EXT.1 Cryptographic Key Storage

FCS_STG_EXT.1.1

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

Application Note: A hardware keystore can be exposed to the TSF through a variety of interfaces, including embedded on the motherboard, USB, microSD, and Bluetooth.

Immutable hardware is considered outside of this requirement and will be covered elsewhere.

If the secure key storage is implemented in software that is protected as required by FCS_STG_EXT.2, the ST author must select software-based. If software-based is selected, the ST author must select all software-based key storage in FCS_STG_EXT.2.1.

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

Validation Guidelines:

Rule #1

FCS_STG_EXT.1.2

The TSF shall be capable of importing keys or secrets into the secure key storage upon request of [**selection**: *the user*, *the administrator*] and [**selection**: *applications running on the TSF*, *no other subjects*].

Application Note: If the ST selects only the user, the ST author must select function 9 in FMT_MOF_EXT.1.1.

FCS_STG_EXT.1.3

The TSF shall be capable of destroying keys or secrets in the secure key storage upon request of [**selection**: *the user*, *the administrator*].

Application Note: If the ST selects the user, the ST author must select function 10 in FMT_MOF_EXT.1.1.

FCS_STG_EXT.1.4

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

Application Note: If the ST selects the user or the administrator, the ST author must also select 34 in FMT_SMF_EXT.1.1. If the ST selects the user, the ST author must select function 34 in FMT_MOF_EXT.1.1.

FCS_STG_EXT.1.5

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

Application Note: If the ST selects the user or the administrator, the ST author must also select function 35 in FMT_SMF_EXT.1.1. If the ST selects only the user, the ST author must select function 35 in FMT_MOF_EXT.1.1.

Evaluation Activities

FCS_STG_EXT.1
The evaluator shall verify that the API documentation provided according to Section 5.2.2 Class

ADV: Development includes the security functions (import, use, and destruction) described in these requirements. The API documentation shall include the method by which applications restrict access to their keys or secrets in order to meet FCS_STG_EXT.1.4.

TSS
The evaluator shall review the TSS to determine that the TOE implements the required secure key storage. The evaluator shall ensure that the TSS contains a description of the key storage mechanism that justifies the selection of "mutable hardware" or "software-based".

Guidance
The evaluator shall review the AGD guidance to determine that it describes the steps needed to import or destroy keys or secrets.

Tests
The evaluator shall test the functionality of each security function:

- *Test FCS_STG_EXT.1:1: The evaluator shall import keys or secrets of each supported type according to the AGD guidance. The evaluator shall write, or the developer shall provide access to, an application that generates a key or secret of each supported type and calls the import functions. The evaluator shall verify that no errors occur during import.*
- *Test FCS_STG_EXT.1:2: The evaluator shall write, or the developer shall provide access to, an application that uses an imported key or secret:*
  - *For RSA, the secret shall be used to sign data.*
  - *For ECDSA, the secret shall be used to sign data.*

  In the future additional types will be required to be tested:
  - *For symmetric algorithms, the secret shall be used to encrypt data.*
  - *For persistent secrets, the secret shall be compared to the imported secret.*

  The evaluator shall repeat this test with the application-imported keys or secrets and a different application's imported keys or secrets. The evaluator shall verify that the TOE requires approval before allowing the application to use the key or secret imported by the user or by a different application:
  - *The evaluator shall deny the approvals to verify that the application is not able to use the key or secret as described.*
  - *The evaluator shall repeat the test, allowing the approvals to verify that the application is able to use the key or secret as described.*

  If the ST author has selected "common application developer", this test is performed by either using applications from different developers or appropriately (according to API documentation) not authorizing sharing.
- *Test FCS_STG_EXT.1:3: The evaluator shall destroy keys or secrets of each supported type according to the AGD guidance. The evaluator shall write, or the developer shall provide access to, an application that destroys an imported key or secret.*

  The evaluator shall repeat this test with the application-imported keys or secrets and a different application's imported keys or secrets. The evaluator shall verify that the TOE requires approval before allowing the application to destroy the key or secret imported by the administrator or by a different application:

  - *The evaluator shall deny the approvals and verify that the application is still able to use the key or secret as described.*
  - *The evaluator shall repeat the test, allowing the approvals and verifying that the application is no longer able to use the key or secret as described.*

  If the ST author has selected "common application developer", this test is performed by either using applications from different developers or appropriately (according to API documentation) not authorizing sharing.

FCS_STG_EXT.2 Encrypted Cryptographic Key Storage

FCS_STG_EXT.2.1

The TSF shall encrypt all DEKs, KEKs, [**assignment**: *any long-term trusted channel key material*] and [**selection**: *all software-based key storage, no other keys*] by KEKs that are [**selection**:

- *Protected by the REK with [**selection**:*
  - *encryption by a REK*
  - *encryption by a KEK chaining from a REK*
  - *encryption by a KEK that is derived from a REK*

]

- *Protected by the REK and the password with [**selection**:*
  - *encryption by a REK and the password-derived KEK*
  - *encryption by a KEK chaining to a REK and the password-derived or biometric-unlocked KEK*
  - *encryption by a KEK that is derived from a REK and the password-derived or biometric-unlocked KEK*

  ]

].

Application Note: The ST author must select all software-based key storage if software-based is selected in FCS_STG_EXT.1.1. If the ST author selects mutable hardware in FCS_STG_EXT.1.1, the secure key storage is not subject to this requirement. REKs are not subject to this requirement.

A REK and the password-derived KEK may be combined to form a combined KEK (as described in FCS_CKM_EXT.3) in order to meet this requirement.

Software-based key storage must be protected by the password or biometric and REK.

All keys must ultimately be protected by a REK. In particular, Figure 3 has KEKs protected according to these requirements: DEK_1 meets the "encryption by a REK and the password-derived KEK" case and would be appropriate for sensitive data, DEK_2 meets the "encryption by a KEK chaining from a REK" case and would not be appropriate for sensitive data, K_1 meets the "encryption by a REK" case and is not considered a sensitive key, and K_2 meets the "encryption by a KEK chaining to a REK and the password-derived or biometric-unlocked KEK" case and is considered a sensitive key.

Long-term trusted channel key material includes Wi-Fi (PSKs), IPsec (PSKs and client certificates) and Bluetooth keys. These keys must not be protected by the password, as they may be necessary in the locked state. For clarity, the ST author must assign any Long-term trusted channel key material supported by the TOE. At a minimum, a TOE must support at least Wi-Fi and Bluetooth keys.

Validation Guidelines:

Rule #1

FCS_STG_EXT.2.2

DEKs, KEKs, [**assignment**: *any long-term trusted channel key material*] and [**selection**: *all software-based key storage, no other keys*] shall be encrypted using one of the following methods: [**selection**:

- *using a SP800-56B key establishment scheme*
- *using AES in the [**selection**: Key Wrap (KW) mode, Key Wrap with Padding (KWP) mode, GCM, CCM, CBC mode]*

].

Application Note: The ST author selects which key encryption schemes are used by the TOE. This requirement refers only to KEKs as defined this PP and does not refer to those KEKs specified in other standards. The ST author must assign the same Long-term trusted channel key material assigned in FCS_STG_EXT.2.1.

Evaluation Activities

FCS_STG_EXT.2
TSS
The evaluator shall review the TSS to determine that the TSS includes key hierarchy description of the protection of each DEK for data-at-rest, of software-based key storage, of long-term trusted channel keys, and of KEK related to the protection of the DEKs, long-term trusted channel keys, and software-based key storage. This description must include a diagram illustrating the key hierarchy implemented by the TOE in order to demonstrate that the implementation meets FCS_STG_EXT.2. The description shall indicate how the functionality described by FCS_RBG.1 is invoked to generate DEKs (FCS_CKM_EXT.2), the key size (FCS_CKM_EXT.2 and FCS_CKM_EXT.3) for each key, how each KEK is formed (generated, derived, or combined according to FCS_CKM_EXT.3), the integrity protection method for each encrypted key (FCS_STG_EXT.3), and the IV generation for each key encrypted by the same KEK (FCS_IV_EXT.1). More detail for each task follows the corresponding requirement.

The evaluator shall also ensure that the documentation of the product's encryption key management is detailed enough that, after reading, the product's key management hierarchy is clear and that it meets the requirements to ensure the keys are adequately protected. The evaluator shall ensure that the documentation includes both an essay and one or more diagrams. Note that this may also be documented as separate proprietary evidence rather than being included in the TSS.

The evaluator shall examine the key hierarchy description in the TSS section to verify that each DEK and software-stored key is encrypted according to FCS_STG_EXT.2.

Guidance
There are no additional Guidance evaluation activities for this component.
There are no guidance evaluation activities for this component.

Tests
There are no test activities for this component.
There are no test evaluation activities for this component.

FCS_STG_EXT.3 Integrity of Encrypted Key Storage

FCS_STG_EXT.3.1

The TSF shall protect the integrity of any encrypted DEKs and KEKs and [**selection**: *long-term trusted channel key material, all software-based key storage, no other keys*] by [**selection**:

- *[**selection**: GCM, CCM, Key Wrap, Key Wrap with Padding] cipher mode for encryption according to FCS_STG_EXT.2*
- *a hash (FCS_COP.1**/Hash**) of the stored key that is encrypted by a key protected by FCS_STG_EXT.2*
- *a keyed hash (FCS_COP.1**/KeyedHash**) using a key protected by a key protected by FCS_STG_EXT.2*
- *a digital signature of the stored key using an asymmetric key protected according to FCS_STG_EXT.2*
- *an immediate application of the key for decrypting the protected data followed by a successful verification of the decrypted data with previously known information*

].

Application Note: The ST author must assign the same Long-term trusted channel key material assigned in FCS_STG_EXT.2.1.

FCS_STG_EXT.3.2

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

Application Note: This requirement is not applicable to derived keys that are not stored. It is not expected that a single key will be protected from corruption by multiple of these methods; however, a product may use one integrity-protection method for one type of key and a different method for other types of keys. The explicit Evaluation Activities for each of the options will be addressed in each of the requirements (FCS_COP.1.1/Hash, FCS_COP.1.1/KeyedHash).

Key Wrapping must be implemented per SP800-38F.

Evaluation Activities

FCS_STG_EXT.3
TSS
The evaluator shall examine the key hierarchy description in the TSS section to verify that each encrypted key is integrity protected according to one of the options in FCS_STG_EXT.3.

The evaluator shall also ensure that the documentation of the product's encryption key management is detailed enough that, after reading, the product's key management hierarchy is clear and that it meets the requirements to ensure the keys are adequately protected. The evaluator shall ensure that the documentation includes both an essay and one or more diagrams. Note that this may also be documented as separate proprietary evidence rather than being included in the TSS.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

5.1.4 Class: User Data Protection (FDP)

A subset of the User Data Protection focuses on protecting Data-At-Rest, namely FDP_DAR_EXT.1 and FDP_DAR_EXT.2. Three levels of data-at-rest protection are addressed: TSF data, Protected Data (and keys), and sensitive data. Table 19 addresses the level of protection required for each level of data-at-rest.

Table 19: Protection of Data Levels

Data Level	Protection Required
TSF Data	TSF data does not require confidentiality, but does require integrity protection. (FPT_TST_EXT.2/PREKERNEL)
Protected Data	Protected data is encrypted while powered off. (FDP_DAR_EXT.1)
Sensitive Data	Sensitive data is encrypted while in the locked state, in addition to while powered off. (FDP_DAR_EXT.2)

All keys, protected data, and sensitive data must ultimately be protected by the REK. Sensitive data must be protected by the password in addition to the REK. In particular, Figure 3 has KEKs protected according to these requirements: DEK_1 would be appropriate for sensitive data, DEK_2 would not be appropriate for sensitive data, K_1 is not considered a sensitive key, and K_2 is considered a sensitive key.

These requirements include a capability for encrypting sensitive data received while in the locked state, which may be considered a separate sub-category of sensitive data. This capability may be met by a key transport scheme (RSA) by using a public key to encrypt the DEK while protecting the corresponding private key with a password-derived or biometric-unlocked KEK.

This capability may also be met by a key agreement scheme. To do so, the device generates a device-wide sensitive data asymmetric pair (the private key of which is protected by a password-derived or biometric-unlocked KEK) and an asymmetric pair for the received sensitive data to be stored. In order to store the sensitive data, the device-wide public key and data private key are used to generate a shared secret, which can be used as a KEK or a DEK. The data private key and shared secret are cleared after the data is encrypted and the data public key stored. Thus, no key material is available in the locked state to decrypt the newly stored data. Upon unlock, the device-wide private key is decrypted and is used with each data public key to regenerate the shared secret and decrypt the stored data. Figure 4, below, illustrates this scheme.

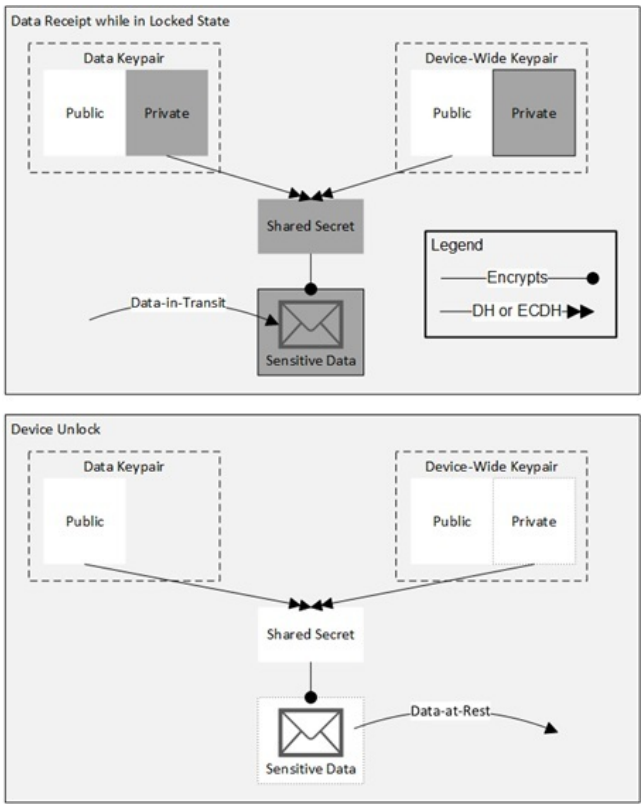


Figure 4: Key Agreement Scheme for Encrypting Received Sensitive Data in the Locked State

FDP_ACF_EXT.1 Access Control for System Services

FDP_ACF_EXT.1.1

The TSF shall provide a mechanism to restrict the system services that are accessible to an application.

Application Note: Examples of system services to which this requirement applies include:

- Obtain data from camera and microphone input devices
- Obtain current device location
- Retrieve credentials from system-wide credential store
- Retrieve contacts list / address book
- Retrieve stored pictures
- Retrieve text messages
- Retrieve emails

- Retrieve device identifier information
- Obtain network access

FDP_ACF_EXT.1.2

The TSF shall provide an access control policy that prevents [**selection**: *application*, *groups of applications*] from accessing [**selection**: *all*, *private*] data stored by other [**selection**: *application*, *groups of applications*]. Exceptions may only be explicitly authorized for such sharing by [**selection**: *the user*, *the administrator*, *a common application developer*, *no one*].

Application Note: Application groups may be designated Enterprise or Personal. Applications installed by the user default to being in the Personal application group unless otherwise designated by the administrator in function 43 of FMT_SMF_EXT.1.1. Applications installed by the administrator default to being in the Enterprise application group (this category includes applications that the user requests the administrator install, for instance by selecting the application for installation through an enterprise application catalog) unless otherwise designated by the administrator in function 43 of FMT_SMF_EXT.1.1. It is acceptable for the same application to have multiple instances installed, each in different application groups. Private data is defined as data that is accessible only by the application that wrote it. Private data is distinguished from data that an application may, by design, write to shared storage areas.

If groups of applications is selected, FDP_ACF_EXT.2 must be included in the ST.

Evaluation Activities

FDP_ACF_EXT.1.1
TSS
The evaluator shall ensure the TSS lists all system services available for use by an application. The evaluator shall also ensure that the TSS describes how applications interface with these system services, and means by which these system services are protected by the TSF.

The TSS shall describe which of the following categories each system service falls in:

1. *No applications are allowed access*
2. *Privileged applications are allowed access*
3. *Applications are allowed access by user authorization*
4. *All applications are allowed access*

Privileged applications include any applications developed by the TSF developer. The TSS shall describe how privileges are granted to third-party applications. For both types of privileged applications, the TSS shall describe how and when the privileges are verified and how the TSF prevents unprivileged applications from accessing those services.

For any services for which the user may grant access, the evaluator shall ensure that the TSS identifies whether the user is prompted for authorization when the application is installed, or during runtime. The evaluator shall ensure that the operational user guidance contains instructions for restricting application access to system services.

Guidance
There are no guidance evaluation activities for this element.

Tests
Evaluation Activity Note: *The following tests require the vendor to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile Device products.*

The evaluator shall write, or the developer shall provide, applications for the purposes of the following tests.

- *Test FDP_ACF_EXT.1.1:1: For each system service to which no applications are allowed access, the evaluator shall attempt to access the system service with a test application and verify that the application is not able to access that system service.*
- *Test FDP_ACF_EXT.1.1:2: For each system service to which only privileged applications are allowed access, the evaluator shall attempt to access the system service with an unprivileged application and verify that the application is not able to access that system service. The evaluator shall attempt to access the system service with a privileged application and verify that the application can access the service.*
- *Test FDP_ACF_EXT.1.1:3: For each system service to which the user may grant access, the evaluator shall attempt to access the system service with a test application. The evaluator shall ensure that either the system blocks such accesses or prompts for user authorization. The prompt for user authorization may occur at runtime or at installation time, and should be consistent with the behavior described in the TSS.*
- *Test FDP_ACF_EXT.1.1:4: For each system service listed in the TSS that is accessible by all applications, the evaluator shall test that an application can access that system service.*

FDP_ACF_EXT.1.2
TSS

The evaluator shall examine the TSS to verify that it describes which data sharing is permitted between applications, which data sharing is not permitted, and how disallowed sharing is prevented. It is possible to select both "applications" and "groups of applications", in which case the TSS is expected to describe the data sharing policies that would be applied in each case.

Guidance
There are no guidance evaluation activities for this element.

Tests

- *Test FDP_ACF_EXT.1.2:1: The evaluator shall write, or the developer shall provide, two applications, one that saves data containing a unique string, and the other, which attempts to access that data. If groups of applications is selected, the applications shall be placed into different groups. If application is selected, the evaluator shall install the two applications. If private is selected, the application shall not write to a designated shared storage area. The evaluator shall verify that the second application is unable to access the stored unique string.*

  If the user is selected, the evaluator shall grant access as the user and verify that the second application is able to access the stored unique string.

  If the administrator is selected, the evaluator shall grant access as the administrator and verify that the second application is able to access the stored unique string.

  If a common application developer is selected, the evaluator shall grant access to an application with a common application developer to the first, and verify that the application is able to access the stored unique string.

FDP_ACF_EXT.2 Access Control for System Resources

This is a selection-based component. Its inclusion depends upon selection from FDP_ACF_EXT.1.2.

FDP_ACF_EXT.2.1

The TSF shall provide a separate [**selection**: *address book*, *calendar*, *keystore*, *account credential database*, *[**assignment**: list of additional resources]*] for each application group and only allow applications within that process group to access the resource. Exceptions may only be explicitly authorized for such sharing by [**selection**: *the user*, *the administrator*, *no one*].

Application Note: If groups of applications is selected in FDP_ACF_EXT.1.2, FDP_ACF_EXT.2 must be included in the ST.

Evaluation Activities

FDP_ACF_EXT.2
TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests
For each selected resource, the evaluator shall cause data to be placed into the Enterprise group's instance of that shared resource. The evaluator shall install an application into the Personal group that attempts to access the shared resource information and verify that it cannot access the information.

FDP_ACF_EXT.3 Security Attribute Based Access Control

This is an objective component.

FDP_ACF_EXT.3.1

The TSF shall enforce an access control policy that prohibits an application from granting both write and execute permission to a file on the device except for [**selection**: *files stored in the application's private data folder*, *no exceptions*].

Evaluation Activities

FDP_ACF_EXT.3

TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests
Evaluation Activity Note: *The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile Device products.*

- *Test FDP_ACF_EXT.3:1: The evaluator shall write, or the developer shall provide, an application that attempts to store a file with both write and execute permissions. If the selection is "no exceptions", then the evaluator shall verify that this action fails and that the permissions on the file are not simultaneously write and execute. If the selection is "application's private data folder", then the evaluator shall ensure that the attempt to store the file is outside of the application's private data folder.*
- *Test FDP_ACF_EXT.3:2: The evaluator shall traverse the file system examining the permission on each TSF file to verify that no file has both write and execute permissions set. If the selection is "application's private data folder", then only files outside of this folder need to be examined by the evaluator for this test.*

FDP_BCK_EXT.1 Application Backup

This is an objective component.

FDP_BCK_EXT.1

The TSF shall provide a mechanism for applications to mark [**selection**: *all application data*, *selected application data*] to be excluded from device backups.

Application Note: Device backups include any mechanism built into the TOE that allows stored application data to be extracted over a physical port or sent over the network, but does not include any functionality implemented by a specific application itself if the application is not included in the TOE. The lack of a public/documented API for performing backups, when a private/undocumented API exists, is not sufficient to meet this requirement.

Evaluation Activities

FDP_BCK_EXT.1
TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests
If all application data is selected, the evaluator shall install an application that has marked all of its application data to be excluded from backups. The evaluator shall cause data to be placed into the application's storage area. The evaluator shall attempt to back up the application data and verify that the backup fails or that the application's data was not included in the backup.

If selected application data is selected, the evaluator shall install an application that has marked selected application data to be excluded from backups. The evaluator shall cause data covered by "selected application data" to be placed into the application's storage area. The evaluator shall attempt to backup that selected application data and verify that either the backup fails or that the selected data is excluded from the backup.

FDP_BLT_EXT.1 Limitation of Bluetooth Device Access

This is an objective component.

FDP_BLT_EXT.1.1

The TSF shall limit the applications that may communicate with a particular paired Bluetooth device.

Application Note: Not every application with privileges to use Bluetooth should be permitted to communicate with every paired Bluetooth device. For example, the TSF may choose to require that only the application that initiated the current

connection may communicate with the device, or it may strictly tie the paired device to the first application that makes a socket connection to the device following initial pairing. Additionally, for more flexibility, the TSF may choose to provide the user with a way to select which applications on the device may communicate with or observe communications with each paired Bluetooth device.

Evaluation Activities

FDP_BLT_EXT.1
TSS
The evaluator shall ensure that the TSS describes the mechanism used to prevent unrestricted access to paired Bluetooth devices (or their communication data) by every application with access to the Bluetooth system service on the TOE. The evaluator shall verify that this method either restricts access to a single application or provides explicit control of the applications that may communicate with the paired Bluetooth device.

Guidance
The evaluator shall verify that the AGD contains the steps to configure which applications are allowed to communicate with a given Bluetooth peripheral.

Tests
The evaluator shall establish a Bluetooth connection with any peripheral. The evaluator shall verify that an application that is allowed to communicate with the Bluetooth peripheral is able to and that an application that is not allowed to communicate with that Bluetooth peripheral is unable to communicate with the peripheral.

FDP_DAR_EXT.1 Protected Data Encryption

FDP_DAR_EXT.1.1

Encryption shall cover all protected data.

Application Note: Protected data is all non-TSF data, including all user or enterprise data. Some or all of this data may be considered sensitive data as well.

FDP_DAR_EXT.1.2

Encryption shall be performed using DEKs with AES in the [**selection**: *CBC*, *GCM*, *XTS*] mode with key size [*256*] bits.

Application Note: IVs must be generated in accordance with FCS_IV_EXT.1.1.

Evaluation Activities

FDP_DAR_EXT.1
TSS
The evaluator shall verify that the TSS section of the ST indicates which data is protected by the DAR implementation and what data is considered TSF data. The evaluator shall ensure that this data includes all protected data.

Guidance
The evaluator shall review the AGD guidance to determine that the description of the configuration and use of the DAR protection does not require the user to perform any actions beyond configuration and providing the authentication credential. The evaluator shall also review the AGD guidance to determine that the configuration does not require the user to identify encryption on a per-file basis.

Tests
Evaluation Activity Note: *The following test requires the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile Device products.*

- *Test FDP_DAR_EXT.1:1: The evaluator shall enable encryption according to the AGD guidance. The evaluator shall create user data (non-system) either by creating a file or by using an application. The evaluator shall use a tool provided by the developer to verify that this data is encrypted when the product is powered off, in conjunction with Test 1 for FIA_UAU_EXT.1.*

FDP_DAR_EXT.2 Sensitive Data Encryption

FDP_DAR_EXT.2.1

The TSF shall provide a mechanism for applications to mark data and keys as sensitive.

Application Note: Data and keys that have been marked as sensitive will be subject to certain restrictions (through other requirements) in both the locked and unlocked states of the Mobile Device. This mechanism allows an application to choose those data and keys under its control to be subject to those requirements.

In the future, this PP may require that all data and key created by applications will default to the "sensitive" marking, requiring an explicit "non-sensitive" marking rather than an explicit "sensitive" marking.

FDP_DAR_EXT.2.2

The TSF shall use an asymmetric key scheme to encrypt and store sensitive data received while the product is locked.

Application Note: Sensitive data is encrypted according to FDP_DAR_EXT.1.2. The asymmetric key scheme must be performed in accordance with FCS_CKM_EXT.7/LOCKED.

The intent of this requirement is to allow the device to receive sensitive data while locked and to store the received data in such a way as to prevent unauthorized parties from decrypting it while in the locked state. If only a subset of sensitive data may be received in the locked state, this subset must be described in the TSS.

Key material must be cleared when no longer needed according to FCS_CKM_EXT.6. For keys (or key material used to derive those keys) protecting sensitive data received in the locked state, "no longer needed" includes "while in the locked state." For example, in the first key scheme, this includes the DEK protecting the received data as soon as the data is encrypted. In the second key scheme this includes the private key for the data asymmetric pair, the generated shared secret, and any generated DEKs. Of course, both schemes require that a private key of an asymmetric pair (the RSA private key and the device-wide private key, respectively) be cleared when transitioning to the locked state.

FDP_DAR_EXT.2.3

The TSF shall encrypt any stored symmetric key and any stored private key of the asymmetric keys used for the protection of sensitive data according to [*FCS_STG_EXT.2.1 selection 2*].

Application Note: Symmetric keys used to encrypt sensitive data while the TSF is in the unlocked state must be encrypted with (or chain to a KEK encrypted with) the REK and password-derived or biometric-unlocked KEK. A stored private key of the asymmetric key scheme for encrypting data in the locked state must be encrypted with (or chain to a KEK encrypted with) the REK and password-derived or biometric-unlocked KEK.

FDP_DAR_EXT.2.4

The TSF shall decrypt the sensitive data that was received while in the locked state upon transitioning to the unlocked state using the asymmetric key scheme and shall re-encrypt that sensitive data using the symmetric key scheme.

Evaluation Activities

FDP_DAR_EXT.2.1
TSS
The evaluator shall verify that the TSS includes a description of which data stored by the TSF (such as by native applications) is treated as sensitive. This data may include all or some user or enterprise data and must be specific regarding the level of protection of email, contacts, calendar appointments, messages, and documents.

The evaluator shall examine the TSS to determine that it describes the mechanism that is provided for applications to use to mark data and keys as sensitive. This description shall also contain information reflecting how data and keys marked in this manner are distinguished from data and keys that are not (for instance, tagging, segregation in a "special" area of memory or container, etc.).

Guidance
There are no guidance evaluation activities for this element.

Tests
The evaluator shall enable encryption of sensitive data and require user authentication according to the AGD guidance. The evaluator shall try to access and create sensitive data (as defined in the ST and either by creating a file or using an application to generate sensitive data) in order to verify that no other user interaction is required.

FDP_DAR_EXT.2.2

TSS
The evaluator shall review the TSS section of the ST to determine that the TSS includes a description of the process of receiving sensitive data while the device is in a locked state. The evaluator shall also verify that the description indicates if sensitive data that may be received in the locked state is treated differently than sensitive data that cannot be received in the locked state. The description shall include the key scheme for encrypting and storing the received data, which must involve an asymmetric key and must prevent the sensitive data-at-rest from being decrypted by wiping all key material used to derive or encrypt the data (as described in the application note). The introduction to this section provides two different schemes that meet the requirements, but other solutions may address this requirement.

Guidance
There are no guidance evaluation activities for this element.

Tests
The evaluator shall perform the tests in FCS_CKM.6 for all key material no longer needed while in the locked state and shall ensure that keys for the asymmetric scheme are addressed in the tests performed when transitioning to the locked state.

FDP_DAR_EXT.2.3
TSS
The evaluator shall verify that the key hierarchy section of the TSS required for FCS_STG_EXT.2.1 includes the symmetric encryption keys (DEKs) used to encrypt sensitive data. The evaluator shall ensure that these DEKs are encrypted by a key encrypted with (or chain to a KEK encrypted with) the REK and password-derived or biometric-unlocked KEK.

The evaluator shall verify that the TSS section of the ST that describes the asymmetric key scheme includes the protection of any private keys of the asymmetric pairs. The evaluator shall ensure that any private keys that are not wiped and are stored by the TSF are stored encrypted by a key encrypted with (or chain to a KEK encrypted with) the REK and password-derived or biometric-unlocked KEK.

The evaluator shall also ensure that the documentation of the product's encryption key management is detailed enough that, after reading, the product's key management hierarchy is clear and that it meets the requirements to ensure the keys are adequately protected. The evaluator shall ensure that the documentation includes both an essay and one or more diagrams. Note that this may also be documented as separate proprietary evidence rather than being included in the TSS.

Guidance
There are no guidance evaluation activities for this element.

Tests
There are no test evaluation activities for this element.

FDP_DAR_EXT.2.4
TSS
The evaluator shall verify that the TSS section of the ST that describes the asymmetric key scheme includes a description of the actions taken by the TSF for the purposes of DAR upon transitioning to the unlocked state. These actions shall minimally include decrypting all received data using the asymmetric key scheme and re-encrypting with the symmetric key scheme used to store data while the device is unlocked.

Guidance
There are no guidance evaluation activities for this element.

Tests
There are no test evaluation activities for this element.

FDP_IFC_EXT.1 Subset Information Flow Control

FDP_IFC_EXT.1.1

The TSF shall [**selection**:

- *provide an interface which allows a VPN client to protect all IP traffic using IPsec*
- *provide a VPN client which can protect all IP traffic using IPsec* ***as defined in the PP-Module for Virtual Private Network (VPN) Clients, version 3.0***

] with the exception of IP traffic needed to manage the VPN connection, and [**selection**: *[**assignment**: traffic needed for correct functioning of the TOE] , no other traffic*], when the VPN is enabled.

Application Note: Typically, the traffic needed to manage the VPN connection is referred to as "Control Plane" traffic; whereas, the IP traffic protected by the

IPsec VPN is referred to as "Data Plane" traffic. All "Data Plane" traffic must flow through the VPN connection and the VPN must not split-tunnel. “IP traffic needed for correct functioning of the TOE” comprises traffic that would prevent the TOE from proper operation if it was either blocked by or routed through the VPN. Enabling the VPN means that the VPN client has been activated by the user. If the VPN tunnel gets interrupted, then no “Data Plane” traffic should be sent without the VPN tunnel being re-established or the user disabling the VPN client.

If no native IPsec client is validated or third-party VPN clients may also implement the required Information Flow Control, the first option must be selected. In these cases, the TOE provides an API to third-party VPN clients that allow them to configure the TOE’s network stack to perform the required Information Flow Control.

The ST author must select the second option if the TSF implements a native VPN client (IPsec in accordance with the PP-Module for Virtual Private Network (VPN) Clients, version 3.0 is selected in FTP_ITC_EXT.1.1). Thus the TSF must be validated against the PP-Module for Virtual Private Network (VPN) Clients, version 3.0 and the ST author must also include FDP_IFC_EXT.1 from the PP-Module for Virtual Private Network (VPN) Clients, version 3.0.

It is optional for the VPN client to be configured to be always-on per FMT_SMF_EXT.1 Function 45. Always-on means the establishment of an IPsec trusted channel to allow any communication by the TSF.

Evaluation Activities

FDP_IFC_EXT.1
TSS
The evaluator shall verify that the TSS section of the ST describes the routing of IP traffic through processes on the TSF when a VPN client is enabled. The evaluator shall ensure that the description indicates which traffic does not go through the VPN and which traffic does. The evaluator shall verify that a configuration exists for each baseband protocol in which only the traffic identified by the ST author as necessary for establishing the VPN connection (IKE traffic and perhaps HTTPS or DNS traffic) or needed for the correct functioning of the TOE is not encapsulated by the VPN protocol (IPsec). The evaluator shall verify that the TSS section describes any differences in the routing of IP traffic when using any supported baseband protocols (e.g. Wi-Fi or, LTE).

Guidance
The evaluator shall verify that one (or more) of the following options is addressed by the documentation:

- *The description above indicates that if a VPN client is enabled, all configurations route all Data Plane traffic through the tunnel interface established by the VPN client.*
- *The AGD guidance describes how the user or administrator can configure the TSF to meet this requirement.*
- *The API documentation includes a security function that allows a VPN client to specify this routing.*

Tests

- *Test FDP_IFC_EXT.1:1: If the ST author identifies any differences in the routing between Wi-Fi and cellular protocols, the evaluator shall repeat this test with a base station implementing one of the identified cellular protocols.*

  Step 1: The evaluator shall enable a Wi-Fi configuration as described in the AGD guidance (as required by FTP_ITC_EXT.1). The evaluator shall use a packet sniffing tool between the wireless access point and an Internet-connected network. The evaluator shall turn on the sniffing tool and perform actions with the device such as navigating to websites, using provided applications, and accessing other Internet resources. The evaluator shall verify that the sniffing tool captures the traffic generated by these actions, turn off the sniffing tool, and save the session data.

  Step 2: The evaluator shall configure an IPsec VPN client that supports the routing specified in this requirement, and if necessary, configure the device to perform the routing specified as described in the AGD guidance. The evaluator shall ensure the test network is capable of sending any traffic identified as exceptions. The evaluator shall turn on the sniffing tool, establish the VPN connection, and perform the same actions with the device as performed in the first step, as well as ensuring that all exception traffic is generated. The evaluator shall verify that the sniffing tool captures traffic generated by these actions, turn off the sniffing tool, and save the session data.

  Step 3: The evaluator shall examine the traffic from both step one and step two to verify that all Data Plane traffic is encapsulated by IPsec, except for any exceptions identified in the SFR (if applicable). For each exception listed in the SFR, the evaluator shall verify that traffic is allowed outside of the VPN tunnel. The evaluator shall examine the Security Parameter Index (SPI) value present in the encapsulated packets captured in Step two from

the TOE to the Gateway and shall verify this value is the same for all actions used to generate traffic through the VPN. Note that it is expected that the SPI value for packets from the Gateway to the TOE is different than the SPI value for packets from the TOE to the Gateway. The evaluator shall be aware that IP traffic on the cellular baseband outside of the IPsec tunnel may be emanating from the baseband processor and shall verify with the manufacturer that any identified traffic is not emanating from the application processor.

Step 4: (Conditional: If ICMP is not listed as part of the IP traffic needed for the correct functioning of the TOE) The evaluator shall perform an ICMP echo from the TOE to the IP address of another device on the local wireless network and shall verify that no packets are sent using the sniffing tool. The evaluator shall attempt to send packets to the TOE outside the VPN tunnel (i.e. not through the VPN gateway), including from the local wireless network, and shall verify that the TOE discards them.

FDP_STG_EXT.1 User Data Storage

FDP_STG_EXT.1.1

The TSF shall provide protected storage for the Trust Anchor Database.

Evaluation Activities

FDP_STG_EXT.1
TSS
The evaluator shall ensure the TSS describes the Trust Anchor Database implemented that contain certificates used to meet the requirements of this PP. This description shall contain information pertaining to how certificates are loaded into the store, and how the store is protected from unauthorized access (for example, UNIX permissions) in accordance with the permissions established in FMT_SMF_EXT.1 and FMT_MOF_EXT.1.1.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FDP_UPC_EXT.1/APPS Inter-TSF User Data Transfer Protection (Applications)

FDP_UPC_EXT.1.1/APPS

The TSF shall provide a means for non-TSF applications executing on the TOE to use [

- *Mutually authenticated TLS as defined in the Functional Package for Transport Layer Security (TLS), version 2.1,*
- *HTTPS,*

*and [**selection**:*

- *mutually authenticated DTLS as defined in the Functional Package for Transport Layer Security (TLS), version 2.1*
- *IPsec as defined in the PP-Module for Virtual Private Network (VPN) Clients, version 3.0*
- *no other protocol*

]] to provide a protected communication channel between the non-TSF application and another IT product that is logically distinct from other communication channels, provides assured identification of its end points, protects channel data from disclosure, and detects modification of the channel data.

Application Note: The intent of this requirement is that one of the selected protocols is available for use by user applications running on the device for use in connecting to distant-end services that are not necessarily part of the enterprise infrastructure. It should be noted that the FTP_ITC_EXT.1 requires that all TSF communications be protected using the protocols indicated in that requirement, so the protocols required by this component ride "on top of" those listed in FTP_ITC_EXT.1.

It should also be noted that some applications are part of the TSF, and FTP_ITC_EXT.1 requires that TSF applications be protected by at least one of the protocols in first selection in FTP_ITC_EXT.1. It is not required to have two different implementations of a protocol, or two different protocols, to satisfy both this requirement (for non-TSF apps) and FTP_ITC_EXT.1 (for TSF apps), as long as the services specified are provided.

The ST author must list which trusted channel protocols are implemented by the Mobile Device for use by non-TSF apps.

The TSF must be validated against FCS_TLSC_EXT requirements from the Functional Package for Transport Layer Security (TLS), version 2.1, with the claims from that package made for those requirements as specified in .
If mutually authenticated DTLS as defined in the Functional Package for Transport Layer Security (TLS), version 2.1 is selected, the TSF must be validated against FCS_DTLS_EXT requirements from the Functional Package for Transport Layer Security (TLS), version 2.1, with the claims from that package made for those requirements as specified in .
If the ST author selects IPsec as defined in the PP-Module for Virtual Private Network (VPN) Clients, version 3.0, the TSF must be validated against the PP-Module for Virtual Private Network (VPN) Clients.

FDP_UPC_EXT.1.2/APPS

The TSF shall permit the non-TSF applications to initiate communication via the trusted channel.

Evaluation Activities

FDP_UPC_EXT.1/APPS
The evaluator shall verify that the API documentation provided according to Section 5.2.2 Class ADV: Development includes the security functions (protection channel) described in these requirements, and verify that the APIs implemented to support this requirement include the appropriate settings/parameters so that the application can both provide and obtain the information needed to assure mutual identification of the endpoints of the communication as required by this component.

TSS
The evaluator shall examine the TSS to determine that it describes that all protocols listed in the TSS are specified and included in the requirements in the ST.

Guidance
The evaluator shall confirm that the operational guidance contains instructions necessary for configuring the protocols selected for use by the applications.

Tests
Evaluation Activity Note: *The following test requires the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile Device products.*

The evaluator shall write, or the developer shall provide access to, an application that requests protected channel services by the TSF. The evaluator shall verify that the results from the protected channel match the expected results according to the API documentation. This application may be used to assist in verifying the protected channel Evaluation Activities for the protocol requirements. The evaluator shall also perform the following tests:

- *Test FDP_UPC_EXT.1/APPS:1: The evaluators shall ensure that the application is able to initiate communications with an external IT entity using each protocol specified in the requirement, setting up the connections as described in the operational guidance and ensuring that communication is successful.*
- *Test FDP_UPC_EXT.1/APPS:2: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data are not sent in plaintext.*

FDP_UPC_EXT.1/BLUETOOTH Inter-TSF User Data Transfer Protection (Bluetooth)

This is an implementation-based component. Its inclusion in depends on whether the TOE implements one or more of the following features:

- ***Bluetooth***

as described in Appendix A.3: Implementation-based Requirements.

FDP_UPC_EXT.1.1/BLUETOOTH

The TSF shall provide a means for non-TSF applications executing on the TOE to use [

- *Bluetooth BR/EDR in accordance with the PP-Module for Bluetooth, version 2.0,*

*and [**selection**:*

- *Bluetooth LE in accordance with the PP-Module for Bluetooth, version 2.0*
- *no other protocol*

] to provide a protected communication channel between the non-TSF application and another IT product that is logically distinct from other communication channels, provides assured identification of its end points, protects channel data from disclosure, and detects modification of the channel data.

Application Note: If the TOE includes Bluetooth hardware, this requirement must be included in the ST. The intent of this requirement is that Bluetooth BR/EDR and optionally Bluetooth LE is available for use by user applications running on the device for use in connecting to distant-end services that are not necessarily part of the enterprise infrastructure. The ST author must list which trusted channel protocols are implemented by the Mobile Device for use by non-TSF apps.

The TSF must be validated against requirements from the PP-Module for Bluetooth, version 2.0. It should be noted that the FTP_ITC_EXT.1 requires that all TSF communications be protected using the protocols indicated in that requirement, so the protocols required by this component ride "on top of" those listed in FTP_ITC_EXT.1.

FDP_UPC_EXT.1.2/BLUETOOTH

The TSF shall permit the non-TSF applications to initiate communication via the trusted channel.

Evaluation Activities

FDP_UPC_EXT.1/BLUETOOTH
The evaluator shall verify that the API documentation provided according to Section 5.2.2 Class ADV: Development includes the security functions (protection channel) described in these requirements, and verify that the APIs implemented to support this requirement include the appropriate settings/parameters so that the application can both provide and obtain the information needed to assure mutual identification of the endpoints of the communication as required by this component.

TSS
The evaluator shall examine the TSS to determine that it describes that all protocols listed in the TSS are specified and included in the requirements in the ST.

Guidance
The evaluator shall confirm that the operational guidance contains instructions necessary for configuring the protocols selected for use by the applications.

Tests
Evaluation Activity Note: *The following test requires the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile Device products.*

The evaluator shall write, or the developer shall provide access to, an application that requests protected channel services by the TSF. The evaluator shall verify that the results from the protected channel match the expected results according to the API documentation. This application may be used to assist in verifying the protected channel Evaluation Activities for the protocol requirements. The evaluator shall also perform the following tests:

- *Test FDP_UPC_EXT.1/BLUETOOTH:1: The evaluators shall ensure that the application is able to initiate communications with an external IT entity using each protocol specified in the requirement, setting up the connections as described in the operational guidance and ensuring that communication is successful.*
- *Test FDP_UPC_EXT.1/BLUETOOTH:2: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data are not sent in plaintext.*

5.1.5 Class: Identification and Authentication (FIA)

FIA_AFL_EXT.1 Authentication Failure Handling

FIA_AFL_EXT.1.1

The TSF shall consider password and [**selection**: *biometric in accordance with the Biometric Enrollment and Verification, version 1.1*, *hybrid*, *no other mechanism*] as critical authentication mechanisms.

Application Note: A critical authentication mechanism is one in which countermeasures are triggered (i.e. wipe of the device) when the maximum number of unsuccessful authentication attempts is exceeded, rendering the other factors unavailable.

If no additional authentication mechanisms are selected in FIA_UAU.5.1, then no other mechanism must be selected. If an additional authentication mechanism is selected in FIA_UAU.5.1, then it must only be selected in FIA_AFL_EXT.1.1 if surpassing the authentication failure threshold for biometric data causes a countermeasure to be triggered regardless of the failure status of the other authentication mechanisms.

If the TOE implements multiple Authentication Factor interfaces (for example, a

DAR decryption interface, a lock screen interface, an auxiliary boot mode interface), this component applies to all available interfaces. For example, a password is a critical authentication mechanism regardless of if it is being entered at the DAR decryption interface or at a lock screen interface.

FIA_AFL_EXT.1.2

The TSF shall detect when a configurable positive integer within [**assignment**: *range of acceptable values for each authentication mechanism*] of [**selection**: *unique*, *non-unique*] unsuccessful authentication attempts occur related to last successful authentication for each authentication mechanism.

Application Note: The positive integers is configured according to FMT_SMF_EXT.1.1 function 2.

An unique authentication attempt is defined as any attempt to verify a password or biometric sample, in which the input is different from a previous attempt. "unique" must be selected if the authentication system increments the counter only for unique unsuccessful authentication attempts. For example, if the same incorrect password is attempted twice the authentication system increments the counter once. "non-unique" must be selected if the authentication system increments the counter for each unsuccessful authentication attempt, regardless of if the input is unique. For example, if the same incorrect password is attempted twice the authentication system increments the counter twice.

If hybrid authentication (i.e. a combination of biometric and pin/password) is supported, a failed authentication attempt can be counted as a single attempt, even if both the biometric and pin/password were incorrect.

If the TOE supports multiple authentication mechanisms per FIA_UAU.5.1, this component applies to all authentication mechanisms. It is acceptable for each authentication mechanism to utilize an independent counter or for multiple authentication mechanisms to utilize a shared counter. The interaction between the authentication factors in regards to the authentication counter must be in accordance with FIA_UAU.5.2.

If the TOE implements multiple Authentication Factor interfaces (for example, a DAR decryption interface, a lock screen interface, an auxiliary boot mode interface), this component applies to all available interfaces. However, it is acceptable for each Authentication Factor interface to be configurable with a different number of unsuccessful authentication attempts.

FIA_AFL_EXT.1.3

The TSF shall maintain the number of unsuccessful authentication attempts that have occurred upon power off.

Application Note: The TOE may implement an Authentication Factor interface that precedes another Authentication Factor interface in the boot sequence (for example, a volume DAR decryption interface which precedes the lock screen interface) before the user can access the device. In this situation, because the user must successfully authenticate to the first interface to access the second, the number of unsuccessful authentication attempts need not be maintained for the second interface.

FIA_AFL_EXT.1.4

When the defined number of unsuccessful authentication attempts has exceeded the maximum allowed for a given authentication mechanism, all future authentication attempts will be limited to other available authentication mechanisms, unless the given mechanism is designated as a critical authentication mechanism.

Application Note: In accordance with FIA_AFL_EXT.1.3, this requirement also applies after the TOE is powered off and powered back on.

FIA_AFL_EXT.1.5

When the defined number of unsuccessful authentication attempts for the last available authentication mechanism or single critical authentication mechanism has been surpassed, the TSF shall perform a wipe of all protected data.

Application Note: Wipe is performed in accordance with FCS_CKM_EXT.5. Protected data is all non-TSF data, including all user or enterprise data. Some or all of this data may be considered sensitive data as well.

If the TOE implements multiple Authentication Factor interfaces (for example, a DAR decryption interface, a lock screen interface, an auxiliary boot mode interface), this component applies to all available interfaces.

FIA_AFL_EXT.1.6

The TSF shall increment the number of unsuccessful authentication attempts prior to notifying the user that the authentication was unsuccessful.

Application Note: This requirement is to ensure that if power is cut to the device directly after an authentication attempt, the counter will be incremented to reflect that attempt.

Evaluation Activities

FIA_AFL_EXT.1
TSS
The evaluator shall ensure that the TSS describes that a value corresponding to the number of unsuccessful authentication attempts since the last successful authentication is kept for each Authentication Factor interface. The evaluator shall ensure that this description also includes if and how this value is maintained when the TOE loses power, either through a graceful powered off or an ungraceful loss of power. The evaluator shall ensure that if the value is not maintained, the interface is after another interface in the boot sequence for which the value is maintained.

If the TOE supports multiple authentication mechanisms, the evaluator shall ensure that this description also includes how the unsuccessful authentication attempts for each mechanism selected in FIA_UAU.5.1 is handled. The evaluator shall verify that the TSS describes if each authentication mechanism utilizes its own counter or if multiple authentication mechanisms utilize a shared counter. If multiple authentication mechanisms utilize a shared counter, the evaluator shall verify that the TSS describes this interaction.

The evaluator shall confirm that the TSS describes how the process used to determine if the authentication attempt was successful. The evaluator shall ensure that the counter would be updated even if power to the device is cut immediately following notifying the TOE user if the authentication attempt was successful or not.

Guidance
The evaluator shall verify that the AGD guidance describes how the administrator configures the maximum number of unique unsuccessful authentication attempts.

Tests

- *Test FIA_AFL_EXT.1:1: The evaluator shall configure the device with all authentication mechanisms selected in FIA_UAU.5.1. The evaluator shall perform the following tests for each available authentication interface:*

  Test 1a: The evaluator shall configure the TOE, according to the AGD guidance, with a maximum number of unsuccessful authentication attempts. The evaluator shall enter the locked state and enter incorrect passwords until the wipe occurs. The evaluator shall verify that the number of password entries corresponds to the configured maximum and that the wipe is implemented.

  Test 1b: [conditional] If the TOE supports multiple authentication mechanisms the previous test shall be repeated using a combination of authentication mechanisms confirming that the critical authentication mechanisms will cause the device to wipe and that when the maximum number of unsuccessful authentication attempts for a non-critical authentication mechanism is exceeded, the device limits authentication attempts to other available authentication mechanisms. If multiple authentication mechanisms utilize a shared counter, then the evaluator shall verify that the maximum number of unsuccessful authentication attempts can be reached by using each individual authentication mechanism and a combination of all authentication mechanisms that share the counter.
- *Test FIA_AFL_EXT.1:2: The evaluator shall repeat test one, but shall power off (by removing the battery, if possible) the TOE between unsuccessful authentication attempts. The evaluator shall verify that the total number of unsuccessful authentication attempts for each authentication mechanism corresponds to the configured maximum and that the critical authentication mechanisms cause the device to wipe. Alternatively, if the number of authentication failures is not maintained for the interface under test, the evaluator shall verify that upon booting the TOE between unsuccessful authentication attempts another authentication factor interface is presented before the interface under test.*

FIA_PMG_EXT.1 Password Management

FIA_PMG_EXT.1.1

The TSF shall support the following for the Password Authentication Factor:

1. Passwords shall be able to be composed of any combination of [**selection**: *upper and lower case letters*, *[**assignment**: a character set of at least 52 characters]*], numbers, and special characters: [**selection**: "*!*", "@", "*#*", "*$*", "*%*", "*^*", "*&*", "***", "*(*", "*)*", *[**assignment**: other characters]*] ;
2. Password length up to [**assignment**: *an integer greater than or equal to 14*] characters shall be supported.

Application Note: While some corporate policies require passwords of 14 characters or better, the use of a REK for DAR protection and key storage protection and the anti-hammer requirement (FIA_TRT_EXT.1) addresses the

threat of attackers with physical access using much smaller and less complex passwords.

The ST author selects the character set: either the upper and lower case Basic Latin letters or another assigned character set containing at least 52 characters. The assigned character set must be well defined: either according to an international encoding standard (such as Unicode) or defined in the assignment by the ST author. The ST author also selects the special characters that are supported by the TOE; they may optionally list additional special characters supported using the assignment.

Evaluation Activities

FIA_PMG_EXT.1
TSS
There are no TSS evaluation activities for this component.

Guidance
The evaluator shall examine the operational guidance to determine that it provides guidance to security administrators on the composition of strong passwords, and that it provides instructions on setting the minimum password length. The evaluator shall also perform the following tests. Note that one or more of these tests can be performed with a single test case.

Tests
- *Test FIA_PMG_EXT.1:1: The evaluator shall compose passwords that either meet the requirements, or fail to meet the requirements, in some way. For each password, the evaluator shall verify that the TOE supports the password. While the evaluator is not required (nor is it feasible) to test all possible compositions of passwords, the evaluator shall ensure that all characters, rule characteristics, and a minimum length listed in the requirement are supported, and justify the subset of those characters chosen for testing.*

FIA_TRT_EXT.1 Authentication Throttling

FIA_TRT_EXT.1.1

The TSF shall limit automated user authentication attempts by [**selection**: *preventing authentication via an external port, enforcing a delay between incorrect authentication attempts*] for all authentication mechanisms selected in FIA_UAU.5.1. The minimum delay shall be such that no more than 10 attempts can be attempted per 500 milliseconds.

Application Note: The authentication throttling applies to all authentication mechanisms selected in FIA_UAU.5.1. The user authentication attempts in this requirement are attempts to guess the Authentication Factor. The developer can implement the timing of the delays in the requirements using unequal or equal timing of delays. The minimum delay specified in this requirement provides defense against brute forcing.

Evaluation Activities

FIA_TRT_EXT.1
TSS
The evaluator shall verify that the TSS describes the method by which authentication attempts are not able to be automated. The evaluator shall ensure that the TSS describes either how the TSF disables authentication via external interfaces (other than the ordinary user interface) or how authentication attempts are delayed in order to slow automated entry and shall ensure that this delay totals at least 500 milliseconds over 10 attempts for all authentication mechanisms selected in FIA_UAU.5.1.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FIA_UAU.5 Multiple Authentication Mechanisms

FIA_UAU.5.1

The TSF shall provide **password and** [**selection**: *biometric in accordance with the Biometric Enrollment and Verification, version 1.1, hybrid, no other mechanism*] to support user authentication.

Application Note: The TSF must support a Password Authentication Factor and may optionally implement a BAF. A hybrid authentication factor is where a user has to submit a combination of PIN/password and biometric sample where both have to pass and if either fails the user is not made aware of which factor failed.

If biometric in accordance with the Biometric Enrollment and Verification, version 1.1 or hybrid is selected, then the TSF must be validated against requirements from the Biometric Enrollment and Verification, version 1.1.

If hybrid is selected, biometric in accordance with the Biometric Enrollment and Verification, version 1.1 does not need to be selected, but should be selected if the biometric authentication can be used independent of the hybrid authentication, i.e. without having to enter a PIN/password.

The Password Authentication Factor is configured according to FIA_PMG_EXT.1.

FIA_UAU.5.2

The TSF shall authenticate any user's claimed identity according to the [**assignment**: *rules describing how the multiple authentication mechanisms provide authentication*].

Application Note: Rules regarding how the authentication factors interact in terms of unsuccessful authentication are covered in FIA_AFL_EXT.1.

Evaluation Activities

FIA_UAU.5
TSS
The evaluator shall ensure that the TSS describes each mechanism provided to support user authentication and the rules describing how the authentication mechanisms provide authentication.

Specifically, for all authentication mechanisms specified in FIA_UAU.5.1, the evaluator shall ensure that the TSS describes the rules as to how each authentication mechanism is used. Example rules are how the authentication mechanism authenticates the user (i.e. how does the TSF verify that the correct password or biometric sample was entered), the result of a successful authentication (i.e. is the user input used to derive or unlock a key) and which authentication mechanism can be used at which authentication factor interfaces (i.e. if there are times, for example, after a reboot, that only specific authentication mechanisms can be used). If multiple BAFs are claimed in FIA_MBV_EXT.1.1 in the Biometric Enrollment and Verification, version 1.1, the interaction between the BAFs must be described. For example, whether the multiple BAFs can be enabled at the same time.

Guidance
The evaluator shall verify that configuration guidance for each authentication mechanism is addressed in the AGD guidance.

Tests
- *Test FIA_UAU.5:1: For each authentication mechanism selected in FIA_UAU.5.1, the evaluator shall enable that mechanism and verify that it can be used to authenticate the user at the specified authentication factor interfaces.*
- *Test FIA_UAU.5:2: For each authentication mechanism rule, the evaluator shall ensure that the authentication mechanisms behave accordingly.*

FIA_UAU.6/CREDENTIAL Re-Authenticating (Credential Change)

FIA_UAU.6.1/CREDENTIAL

The TSF shall re-authenticate the user **via the Password Authentication Factor** under the conditions [*attempted change to any supported authentication mechanisms*].

Application Note: The password authentication factor must be entered before either the password or biometric authentication factor, if selected in FIA_UAU.5.1, can be changed.

Evaluation Activities

FIA_UAU.6.1/CREDENTIAL
TSS
There are no TSS evaluation activities for this element.

Guidance
There are no guidance evaluation activities for this element.

Tests

- *Test FIA_UAU.6.1/CREDENTIAL:1: The evaluator shall configure the TSF to use the Password Authentication Factor according to the AGD guidance. The evaluator shall change Password Authentication Factor according to the AGD guidance and verify that the TSF requires the entry of the Password Authentication Factor before allowing the factor to be changed.*
- *Test FIA_UAU.6.1/CREDENTIAL:2: [conditional] If biometric in accordance with the Biometric Enrollment and Verification, version 1.1 is selected in FIA_UAU.5.1, for each BAF claimed in FIA_MBV_EXT.1.1 in the Biometric Enrollment and Verification, version 1.1 the evaluator shall configure the TSF to use the BAF, which includes configuring the Password Authentication Factor, according to the AGD guidance. The evaluator shall change the BAF according to the AGD guidance and verify that the TSF requires the entry of the Password Authentication Factor before allowing the BAF to be changed.*
- *Test FIA_UAU.6.1/CREDENTIAL:3: [conditional] If hybrid is selected in FIA_UAU.5.1, the evaluator shall configure the TSF to use the BAF and PIN or password, which includes configuring the Password Authentication Factor, according to the AGD guidance. The evaluator shall change the BAF and PIN according to the AGD guidance and verify that the TSF requires the entry of the Password Authentication Factor before allowing the factor to be changed.*

FIA_UAU.6/LOCKED Re-Authenticating (TSF Lock)

FIA_UAU.6.1/LOCKED

The TSF shall re-authenticate the user **via an authentication factor defined in FIA_UAU.5.1** under the conditions **TSF-initiated lock, user-initiated lock, [assignment: *other conditions*]** .

Application Note: Depending on the selections made in FIA_UAU.5.1, either the password (at a minimum), biometric authentication or hybrid authentication mechanisms can be used to unlock the device. TSF-initiated and user-initiated locking is described in FTA_SSL_EXT.1.

Evaluation Activities

FIA_UAU.6.1/LOCKED
TSS
There are no TSS evaluation activities for this element.

Guidance
There are no guidance evaluation activities for this element.

Tests

- *Test FIA_UAU.6.1/LOCKED:1: The evaluator shall configure the TSF to transition to the locked state after a time of inactivity (FMT_SMF_EXT.1) according to the AGD guidance. The evaluator shall wait until the TSF locks and then verify that the TSF requires the entry of the Password Authentication Factor before transitioning to the unlocked state.*
- *Test FIA_UAU.6.1/LOCKED:2: [conditional] If biometric in accordance with the Biometric Enrollment and Verification, version 1.1 is selected in FIA_UAU.5.1, for each BAF claimed in FIA_MBV_EXT.1.1 in the Biometric Enrollment and Verification, version 1.1 the evaluator shall repeat Test 1 verifying that the TSF requires the entry of the BAF before transitioning to the unlocked state.*
- *Test FIA_UAU.6.1/LOCKED:3: [conditional] If hybrid is selected in FIA_UAU.5.1, the evaluator shall repeat Test 1 verifying that the TSF requires the entry of the BAF and PIN/password before transitioning to the unlocked state.*
- *Test FIA_UAU.6.1/LOCKED:4: The evaluator shall configure user-initiated locking according to the AGD guidance. The evaluator shall lock the TSF and then verify that the TSF requires the entry of the Password Authentication Factor before transitioning to the unlocked state.*
- *Test FIA_UAU.6.1/LOCKED:5: [conditional] If biometric in accordance with the Biometric Enrollment and Verification, version 1.1 is selected in FIA_UAU.5.1, for each BAF claimed in FIA_MBV_EXT.1.1 in the Biometric Enrollment and Verification, version 1.1 the evaluator shall repeat Test FIA_UAU.6.1/LOCKED:4 verifying that the TSF requires the entry of the BAF before transitioning to the unlocked state.*
- *Test FIA_UAU.6.1/LOCKED:6: [conditional] If hybrid is selected in FIA_UAU.5.1, the evaluator shall repeat Test FIA_UAU.6.1/LOCKED:4 verifying that the TSF requires the entry of the BAF and PIN/password before transitioning to the unlocked state.*

FIA_UAU.7 Protected Authentication Feedback

FIA_UAU.7.1

The TSF shall provide only [*obscured feedback to the device's display*] to the user while the authentication is in progress.

Application Note: This applies to all authentication methods specified in FIA_UAU.5.1. The TSF may briefly (1 second or less) display each character or provide an option to allow the user to unmask the password; however, the

password must be obscured by default.

If biometric in accordance with the Biometric Enrollment and Verification, version 1.1 is selected in FIA_UAU.5.1, the TSF must not display sensitive information regarding any BAF that could aid an adversary in identifying or spoofing the respective biometric characteristics of a given human user. While it is true that biometric samples, by themselves, are not secret, the analysis performed by the respective biometric algorithms, as well as output data from these biometric algorithms, is considered sensitive and must be kept secret. Where applicable, the TSF must not reveal or make public the reasons for authentication failure.

Evaluation Activities

FIA_UAU.7
TSS
The evaluator shall ensure that the TSS describes the means of obscuring the authentication entry, for all authentication methods specified in FIA_UAU.5.1.

Guidance
The evaluator shall verify that any configuration of this requirement is addressed in the AGD guidance and that the password is obscured by default.

Tests
- *Test FIA_UAU.7:1: The evaluator shall enter passwords on the device, including at least the Password Authentication Factor at lock screen, and verify that the password is not displayed on the device.*
- *Test FIA_UAU.7:2: [conditional] If biometric in accordance with the Biometric Enrollment and Verification, version 1.1 is selected in FIA_UAU.5.1, for each BAF claimed in FIA_MBV_EXT.1.1 in the Biometric Enrollment and Verification, version 1.1 the evaluator shall authenticate by producing a biometric sample at lock screen. As the biometric algorithms are performed, the evaluator shall verify that sensitive images, audio, or other information identifying the user are kept secret and are not revealed to the user. Additionally, the evaluator shall produce a biometric sample that fails to authenticate and verify that the reasons for authentication failure (user mismatch, low sample quality, etc.) are not revealed to the user. It is acceptable for the BAF to state that it was unable to physically read the biometric sample, for example, if the sensor is unclean or the biometric sample was removed too quickly. However, specifics regarding why the presented biometric sample failed authentication shall not be revealed to the user.*

FIA_UAU_EXT.1 Authentication for Cryptographic Operation

FIA_UAU_EXT.1.1

The TSF shall require the user to present the Password Authentication Factor prior to decryption of protected data and encrypted DEKs, KEKs and [**selection**: *long-term trusted channel key material*, *all software-based key storage*, *no other keys*] at startup.

Application Note: The intent of this requirement is to prevent decryption of protected data before the user has authorized to the device using the Password Authentication Factor. The Password Authentication Factor is also required in order derive the key used to decrypt sensitive data, which includes software-based secure key storage.

Evaluation Activities

FIA_UAU_EXT.1
TSS
The evaluator shall verify that the TSS section of the ST describes the process for decrypting protected data and keys. The evaluator shall ensure that this process requires the user to enter a Password Authentication Factor and, in accordance with FCS_CKM_EXT.3, derives a KEK, which is used to protect the software-based secure key storage and (optionally) DEKs for sensitive data, in accordance with FCS_STG_EXT.2.

Guidance
There are no guidance evaluation activities for this component.

Tests
The following tests may be performed in conjunction with FDP_DAR_EXT.1 and FDP_DAR_EXT.2.

Evaluation Activity Note: *The following test require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile Device products.*

- *Test FIA_UAU_EXT.1:1: The evaluator shall enable encryption of protected data and require user authentication according to the AGD guidance. The evaluator shall write, or the developer shall provide access to, an application that includes a unique string treated as protected data.*

  The evaluator shall reboot the device, use a tool provided by developer to search for the unique string within the application data, and verify that the unique string cannot be found. The evaluator shall enter the Password Authentication Factor to access full device functionality, use a tool provided by the developer to access the unique string within the application data, and verify that the unique string can be found.
- *Test FIA_UAU_EXT.1:2: [conditional] The evaluator shall require user authentication according to the AGD guidance. The evaluator shall store a key in the software-based secure key storage.*

  The evaluator shall lock the device, use a tool provided by developer to access the key within the stored data, and verify that the key cannot be retrieved or accessed. The evaluator shall enter the Password Authentication Factor to access full device functionality, use a tool provided by developer to access the key, and verify that the key can be retrieved or accessed.
- *Test FIA_UAU_EXT.1:3: [conditional] The evaluator shall enable encryption of sensitive data and require user authentication according to the AGD guidance. The evaluator shall write, or the developer shall provide access to, an application that includes a unique string treated as sensitive data.*

  The evaluator shall lock the device, use a tool provided by developer to attempt to access the unique string within the application data, and verify that the unique string cannot be found. The evaluator shall enter the Password Authentication Factor to access full device functionality, use a tool provided by developer to access the unique string within the application data, and verify that the unique string can be retrieved.

FIA_UAU_EXT.2 Timing of Authentication

FIA_UAU_EXT.2.1

The TSF shall allow [**selection**: ***[assignment**: list of actions]* , *no actions*] on behalf of the user to be performed before the user is authenticated.

FIA_UAU_EXT.2.2

The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

Application Note: The security relevant actions allowed by unauthorized users in locked state must be listed. At a minimum the actions that correspond to the functions available to the user in FMT_SMF_EXT.1 and are allowed by unauthorized users in locked state should be listed. For example, if the user can enable/disable the camera per function 5 of FMT_SMF_EXT.1 and unauthorized users can take a picture when the device is in locked state, this action must be listed.

Evaluation Activities

FIA_UAU_EXT.2
TSS
The evaluator shall verify that the TSS describes the actions allowed by unauthorized users in the locked state.

Guidance
There are no guidance evaluation activities for this component.

Tests
The evaluator shall attempt to perform some actions not listed in the selection while the device is in the locked state and verify that those actions do not succeed.

FIA_UAU_EXT.4 Secondary User Authentication

This is an optional component. However, applied modules or packages might redefine it as mandatory.

FIA_UAU_EXT.4.1

The TSF shall provide a secondary authentication mechanism for accessing Enterprise applications and resources. The secondary authentication mechanism shall control access to the Enterprise application and shared resources and shall be incorporated into the encryption of protected and sensitive data belonging to Enterprise applications and shared resources.

Application Note: For the BYOD use case, Enterprise applications and data must be protected using a different password than the user authentication to gain access to the personal applications and data, if configured.

This requirement must be included in the ST if the TOE implements a container solution, in which there is a separate authentication, to separate user and Enterprise applications and resources.

FIA_UAU_EXT.4.2

The TSF shall require the user to present the secondary authentication factor prior to decryption of Enterprise application data and Enterprise shared resource data.

Application Note: The intent of this requirement is to prevent decryption of protected Enterprise application data and Enterprise shared resource data before the user has authenticated to the device using the secondary authentication factor. Enterprise shared resource data consists of the FDP_ACF_EXT.2.1 selections.

Evaluation Activities

FIA_UAU_EXT.4.1
TSS
There are no TSS evaluation activities for this element.

Guidance
There are no guidance evaluation activities for this element.

Tests
The Evaluation Activities for any selected requirements related to device authentication must be separately performed for the secondary authentication mechanism (in addition to activities performed for the primary authentication mechanism). The requirements are: FIA_UAU.6/CREDENTIAL, FIA_UAU.6/LOCKED, FIA_PMG_EXT.1, FIA_TRT_EXT.1, FIA_UAU.7, FIA_UAU_EXT.2, FTA_SSL_EXT.1, FCS_STG_EXT.2, FMT_SMF_EXT.1/FMT_MOF_EXT.1 #1, #2, #8, #21, and #36.

Additionally, FIA_AFL_EXT.1 must be met, except that in FIA_AFL_EXT.1.2 the separate test is performed with the text "wipe of all protected data" changed to "wipe of all Enterprise application data and all Enterprise shared resource data."

FIA_UAU_EXT.4.2
TSS
The evaluator shall verify that the TSS section of the ST describes the process for decrypting Enterprise application data and shared resource data. The evaluator shall ensure that this process requires the user to enter an Authentication Factor and, in accordance with FCS_CKM_EXT.3, derives a KEK which is used to protect the software-based secure key storage and (optionally) DEKs for sensitive data, in accordance with FCS_STG_EXT.2.

Guidance
There are no guidance evaluation activities for this element.

Tests
There are no test evaluation activities for this element.

FIA_X509_EXT.6 Request Validation of Certificates

FIA_X509_EXT.6.1

The TSF shall provide a certificate validation service to applications.

FIA_X509_EXT.6.2

The TSF shall respond to the requesting application with the success or failure of the validation.

Application Note: In order to comply with all of the rules in FIA_X509_EXT.1 as defined in Functional Package for X.509, version 1.0, multiple API calls may be required; all of these calls should be clearly documented.

Evaluation Activities

FIA_X509_EXT.6

The evaluator shall verify that the API documentation provided according to Section 5.2.2 Class ADV: Development includes the security function (certificate validation) described in this requirement. This documentation shall be clear as to which results indicate success and failure.

TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests
The evaluator shall write, or the developer shall provide access to, an application that requests certificate validation by the TSF. The evaluator shall verify that the results from the validation match the expected results according to the API documentation. This application may be used to verify that import, removal, modification, and validation are performed correctly according to the tests required by FDP_STG_EXT.1, FTP_ITC_EXT.1, FMT_SMF_EXT.1, and FIA_X509_EXT.1 as defined in Functional Package for X.509, version 1.0.

5.1.6 Class: Security Management (FMT)

Both the user and the administrator may manage the TOE. This administrator is likely to be acting remotely and could be the Mobile Device Management (MDM) Administrator acting through an MDM Agent.

The Administrator is responsible for management activities, including setting the policy that is applied by the enterprise on the Mobile Device. These management functions are likely to be a different set than those management functions provided to the user. Management functions that are provided to the user and not the administrator are listed in FMT_MOF_EXT.1.1. Management functions for which the administrator may adopt a policy that restricts the user from performing that function are listed in FMT_MOF_EXT.1.2.

Table 20 compares the management functions required by this Protection Profile in the following three requirements (FMT_MOF_EXT.1.1, FMT_MOF_EXT.1.2, and FMT_SMF_EXT.1).

FMT_MOF_EXT.1 Management of Security Functions Behavior

FMT_MOF_EXT.1.1

The TSF shall restrict the ability to perform the functions [*in column 4 of Table 20*] to the user.

Application Note: The functions that have an "M" in the fourth column are mandatory for this component, thus are restricted to the user, meaning that the administrator cannot manage those functions. The functions that have an "O" in the fourth column are optional and may be selected; and those functions with a "-" are not applicable and may not be selected. The ST author should select those security management functions that only the user may perform (i.e. the ones the administrator may not perform).

The ST author may not select the same function in both FMT_MOF_EXT.1.1 and FMT_MOF_EXT.1.2. A function cannot contain an "M" in both column 4 and column 6.

The ST author may use a table in the ST, indicating with clear demarcations (to be accompanied by an index) those functions that are restricted to the user (column 4). The ST author should iterate a row to indicate any variations in the selectable sub-functions or assigned values with respect to the values in the columns.

For functions that are mandatory, any sub-functions not in a selection are also mandatory and any assignments must contain at least one assigned value. For non-selectable sub-functions in an optional function, all sub-functions outside a selection must be implemented in order for the function to be listed.

FMT_MOF_EXT.1.2

The TSF shall restrict the ability to perform the functions [*in column 6 of Table 20*] to the administrator when the device is enrolled and according to the administrator-configured policy.

Application Note: As long as the device is enrolled in management, the administrator (of the enterprise) must be guaranteed that minimum security functions of the enterprise policy are enforced. Further restrictive policies can be applied at any time by the user on behalf of the user or other administrators.

The functions that have an "M" in the sixth column are mandatory for this component; the functions that have an "O" in the sixth column are optional and may be selected; and those functions with a "-" in the sixth are not applicable and may not be selected.

The ST author may not select the same function in both FMT_MOF_EXT.1.1 and FMT_MOF_EXT.1.2.

The ST author should select those security management functions that the administrator may restrict. The ST author may use a table in the ST, indicating with clear demarcations (to be accompanied by an index) those functions that are and are not implemented with APIs for the administrator (as in column 5). Additionally, the ST author should demarcate which functions the user is prevented from accessing or performing (as in column 6). The ST author should iterate a row to indicate any variations in the selectable sub-functions or assigned values with respect to the values in the columns.

For functions that are mandatory, any sub-functions not in a selection are also mandatory and any assignments must contain at least one assigned value. For non-selectable sub-functions in an optional function, all sub-functions outside the selection must be implemented in order for the function to be listed.

Evaluation Activities

FMT_MOF_EXT.1.1
TSS
The evaluator shall verify that the TSS describes those management functions that may only be performed by the user and confirm that the TSS does not include an Administrator API for any of these management functions. This activity will be performed in conjunction with FMT_SMF_EXT.1.

Guidance
There are no guidance evaluation activities for this element.

Tests
There are no test evaluation activities for this element.

FMT_MOF_EXT.1.2
TSS
The evaluator shall verify that the TSS describes those management functions that may be performed by the Administrator, to include how the user is prevented from accessing, performing, or relaxing the function (if applicable), and how applications/APIs are prevented from modifying the Administrator configuration. The TSS also describes any functionality that is affected by administrator-configured policy and how. This activity will be performed in conjunction with FMT_SMF_EXT.1.

Guidance
There are no guidance evaluation activities for this element.

Tests

- *Test FMT_MOF_EXT.1.2:1: The evaluator shall use the test environment to deploy policies to Mobile Devices.*
- *Test FMT_MOF_EXT.1.2:2: The evaluator shall create policies which collectively include all management functions which are controlled by the (enterprise) administrator and cannot be overridden/relaxed by the user as defined in FMT_MOF_EXT.1.2. The evaluator shall apply these policies to devices, attempt to override/relax each setting both as the user (if a setting is available) and as an application (if an API is available), and ensure that the TSF does not permit it. Note that the user may still apply a more restrictive policy than that of the administrator.*
- *Test FMT_MOF_EXT.1.2:3: Additional testing of functions provided to the administrator are performed in conjunction with the testing activities for FMT_SMF_EXT.1.1.*

FMT_SMF_EXT.1 Specification of Management Functions

FMT_SMF_EXT.1.1

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

Table 20: Management Functions

Status Markers:
M - Mandatory
O - Optional/Objective

#	Management Function	Impl.	User Only	Admin	Admin Only
1	configure password policy: • Minimum password length • Minimum password complexity	M	-	M	M

	• Maximum password lifetime				
2	configure session locking policy: • Screen-lock enabled/disabled • Screen lock timeout • Number of authentication failures	M	-	M	M
3	enable/disable the VPN protection: • Across device [**selection**: • *on a per-app basis* • *on a per-group of applications processes basis* • *no other method*]	M	O	O	O
4	enable/disable [**assignment**: *list of all radios*]	M	O	O	O
5	enable/disable [**assignment**: *list of audio or visual collection devices*] : • Across device [**selection**: • *on a per-app basis* • *on a per-group of applications processes basis* • *no other method*]	M	O	O	O
6	transition to the locked state	M	-	M	-
7	TSF wipe of protected data	M	-	M	-
8	configure application installation policy by [**selection**: • *restricting the sources of applications* • *specifying a set of allowed applications based on [**assignment**: application characteristics] (an application allowlist)* • *denying installation of applications*]	M	-	M	M
9	import keys or secrets into the secure key storage	M	O	O	-
10	destroy imported keys or secrets and [**selection**: *no other keys or secrets, [**assignment**: list of other categories of keys or secrets]*] in the secure key storage	M	O	O	-
11	import X.509v3 certificates into the Trust Anchor Database	M	-	M	O
12	remove imported X.509v3 certificates and [**selection**: *no other X.509v3 certificates, [**assignment**: list of other categories of X.509v3 certificates]*] in the Trust Anchor Database	M	O	O	-
13	enroll the TOE in management	M	O	O	O
14	remove applications	M	-	M	O
15	update system software	M	-	M	O
16	install applications	M	-	M	O
17	remove Enterprise applications	M	-	M	-
18	enable/disable display notification in the locked state of: [**selection**: • *email notifications* • *calendar appointments* • *contact associated with phone call notification* • *text message notification*	M	O	O	O

- *[**assignment**: other application-based notifications]*
- *all notifications*

]

19	enable data-at rest protection	M	O	O	O
20	enable removable media’s data-at-rest protection	M	O	O	O
21	enable/disable location services: • Across device [**selection**: • *on a per-app basis* • *on a per-group of applications processes basis* • *no other method*]	M	O	O	O
22	enable/disable the use of [**selection**: *Biometric Authentication Factor, Hybrid Authentication Factor*]	O	O	O	O
23	configure whether to allow or disallow establishment of [**assignment**: *configurable trusted channel in FTP_ITC_EXT.1.1 or FDP_UPC_EXT.1.1/APPS*] if the peer or server certificate is deemed invalid.	O	O	O	O
24	enable/disable all data signaling over [**assignment**: *list of externally accessible hardware ports*]	O	O	O	O
25	enable/disable [**assignment**: *list of protocols where the device acts as a server*]	O	O	O	O
26	enable/disable developer modes	O	O	O	O
27	enable/disable bypass of local user authentication	O	O	O	O
28	wipe Enterprise data	O	O	O	-
29	approve [**selection**: *import, removal*] by applications of X.509v3 certificates in the Trust Anchor Database	O	O	O	O
30	configure whether to allow or disallow establishment of a trusted channel if the TSF cannot establish a connection to determine the validity of a certificate	O	O	O	O
31	enable/disable the cellular protocols used to connect to cellular network base stations	O	O	O	O
32	read audit logs kept by the TSF	O	O	O	-
33	configure [**selection**: *certificate, public-key*] used to validate digital signature on applications	O	O	O	O
34	approve exceptions for shared use of keys or secrets by multiple applications	O	O	O	O
35	approve exceptions for destruction of keys or secrets by applications that did not import the key or secret	O	O	O	O
36	configure the unlock banner	M	-	O	O
37	configure the auditable items	O	-	O	O
38	retrieve TSF-software integrity verification values	O	O	O	O
39	enable/disable [**selection**: • *USB mass storage mode* • *USB data transfer without user authentication*	O	O	O	O

	• *USB data transfer without authentication of the connecting system*]				
40	enable/disable backup of [**selection**: *all applications, selected applications, selected groups of applications, configuration data*] to [**selection**: *locally connected system, remote system*]	O	O	O	O
41	enable/disable [**selection**: • *Hotspot functionality authenticated by [**selection**: pre-shared key, passcode, no authentication]* • *USB tethering authenticated by [**selection**: pre-shared key, passcode, no authentication]*]	O	O	O	O
42	approve exceptions for sharing data between [**selection**: *applications, groups of applications*]	O	O	O	O
43	place applications into application groups based on [**assignment**: *enterprise configuration settings*]	O	O	O	O
44	unenroll the TOE from management	O	O	O	O
45	enable/disable the Always On VPN protection: • Across device [**selection**: • *on a per-app basis* • *on a per-group of applications processes basis* • *no other method*]	O	O	O	O
46	revoke Biometric template	O	O	O	O
47	[**assignment**: *list of other management functions to be provided by the TSF*]	O	O	O	O

].

Application Note: Table 20 compares the management functions required by this Protection Profile.

The first column lists the management functions identified in the PP.

In the following columns:

- ‘M’ means Mandatory
- ‘O’ means Optional/Objective
- '-' means that no value (M or O) can be assigned

The third column ("Impl.") indicates whether the function is to be implemented. The ST author should select which Optional functions are implemented.

The fourth column ("User Only") indicates functions that are to be restricted to the user (i.e. not available to the administrator).

The fifth column ("Admin") indicates functions that are available to the administrator. The functions restricted to the user (column 4) cannot also be available to the administrator. Functions available to the administrator can still be available to the user, as long as the function is not restricted to the administrator (column 6). Thus, if the TOE must offer these functions to the administrator to perform, the fifth column must be selected.

The sixth column (FMT_MOF_EXT.1.2) indicates whether the function is to be restricted to the administrator when the device is enrolled and the administrator applies the indicated policy. If the function is restricted to the administrator the function is not available to the user. This does not prevent the user from modifying a setting to make the function stricter, but the user cannot undo the configuration enforced by the administrator.

The ST author may use a table in the ST, listing only those functions that are

implemented. For functions that are mandatory, any sub-functions not in a selection are also mandatory and any assignments must contain at least one assigned value. For functions that are optional and contain an assignment or selection, at least one value must be assigned/selected to be included in the ST. For non-selectable sub-functions in an optional function, all sub-functions must be implemented in order for the function to be included. For functions with a "per-app basis" sub function and an assignment, the ST author must indicate which assigned features are manageable on a per-app basis and which are not by iterating the row.

Function-specific Application Notes:

Functions 3 , 5 , 21 , 5 , and 21 must be implemented on a device-wide basis but may also be implemented on a per-app basis or on a per-group of applications basis in which the configuration includes the list of applications or groups of applications to which the enable/disable applies.

Function 3 addresses enabling and disabling the IPsec VPN only. The configuration of the VPN Client itself (with information such as VPN Gateway, certificates, and algorithms) is addressed by the PP-Module for Virtual Private Network (VPN) Clients, version 3.0. The administrator options should only be listed if the administrator can remotely enable/disable the VPN connection.

Function 3 optionally allows the VPN to be configured per-app or per-groups of apps. If this configuration is selected, it does not void FDP_IFC_EXT.1. Instead FDP_IFC_EXT.1 is applied to the application or group of applications the VPN is applied to. In other words, all traffic destined for the VPN-enabled application or group of applications, must travel through the VPN, but traffic not destined for that application or group of applications can travel outside the VPN. When the VPN is configured across the device FDP_IFC_EXT.1 applies to all traffic and the VPN must not split tunnel.

The assignment in function 4 consists of all radios present on the TSF, such as Wi-Fi, cellular, NFC, Bluetooth BR/EDR, and Bluetooth LE, which can be enabled and disabled. In the future, if both Bluetooth BR/EDR and Bluetooth LE are supported, they will be required to be enabled and disabled separately. Disablement of the cellular radio does not imply that the radio may not be enabled in order to place emergency phone calls; however, it is not expected that a device in "airplane mode", where all radios are disabled, will automatically (without authorization) turn on the cellular radio to place emergency calls.

The assignment in function 5 consists of at least one audio or visual device, such as camera and microphone, which can be enabled and disabled by either the user or administrator. Disablement of the microphone does not imply that the microphone may not be enabled in order to place emergency phone calls. If certain devices are able to be restricted to the enterprise (either device-wide, per-app or per-group of applications) and others are able to be restricted to users, then this function should be iterated in the table with the appropriate table entries.

Regarding functions 4 and 5, disablement of a particular radio or audio/visual device must be effective as soon as the TOE has power. Disablement must also apply when the TOE is booted into auxiliary boot modes, for example, associated with updates or backup. If the TOE supports states in which security management policy is inaccessible, for example, due to data-at-rest protection, it is acceptable to meet this requirement by ensuring that these devices are disabled by default while in these states. That these devices are disabled during auxiliary boot modes does not imply that the device (particularly the cellular radio) may not be enabled in order to perform emergency phone calls.

Wipe of the TSF (function 7) is performed according to FCS_CKM_EXT.5. Protected data is all non-TSF data, including all user or enterprise data. Some or all of this data may be considered sensitive data as well.

The selection in function 8 allows the ST author to select which mechanisms are available to the administrator through the MDM Agent to restrict the applications which the user may install. The ST author must state if application allowlist is applied device-wide or if it can be specified to apply to either the Enterprise or Personal applications.

- If the administrator can restrict the sources from which applications can be installed, the ST author selects "restricting the sources of applications".
- If the administrator can specify a allowlist of allowed applications, the ST author selects "specifying a set of allowed applications based on [**assignment**: *application characteristics*] (an application allowlist)". The ST author should list any application characteristics (e.g. name, version, or developer) based on which the allowlist can be formed.
- If the administrator can prevent the user from installing additional applications, the ST author selects "denying installation of applications".

In the future, function 12 may require destruction or disabling of any default trusted CA certificates, excepting those CA certificates necessary for continued operation of the TSF, such as the developer’s certificate. At this time, the ST

author must indicate in the assignment whether pre-installed or any other category of X.509v3 certificates may be removed from the Trust Anchor Database.

For function 13, the enrollment function may be installing an MDM agent and includes the policies to be applied to the device. It is acceptable for the user approval notice to require the user to intentionally opt to view the policies (for example, by "tapping" on a "View" icon) rather than listing the policies in full in the notice.

For function 15, the administrator capability to update the system software may be limited to causing a prompt to the user to update rather than the ability to initiate the update itself. As the administrator is likely to be acting remotely, he/she would be unaware of inopportune situations, such as low power, which may cause the update to fail and the device to become inoperable. The user can refuse to accept the update in such situations. It is expected that system architects will be cognizant of this limitation and will enforce network access controls in order to enforce enterprise-critical updates.

Function 16 addresses both installation and update. This protection profile does not distinguish between installation and update of applications because mobile devices typically completely overwrite the previous installation with a new installation during an application update.

For function 17, "Enterprise applications" are those applications that belong to the Enterprise application group. Applications installed by the enterprise administrator (including automatic installation by the administrator after being requested by the user from a catalog of enterprise applications) are by default placed in the Enterprise application group unless an exception has been made in function 43 of FMT_SMF_EXT.1.1.

If the display of notifications in the locked state is supported, the configuration of these notifications (function 18) must be included in the selection.

Function 19 must be included in the selection if data-at-rest protection is not natively enabled.

Function 20 is implicitly met if the TSF does not support removable media.

For function 21, location services include location information gathered from GPS, cellular, and Wi-Fi.

Function 22 must be included in the ST if the TOE contains a BAF. This selection must correspond with the selection made in FIA_UAU.5.1. If biometric in accordance with the Biometric Enrollment and Verification, version 1.1 is selected in FIA_UAU.5.1, "Biometric Authentication Factor" must be selected and the user or admin must have the option to disable the use of it. If multiple BAFs are claimed in FIA_MBV_EXT.1.1 in the Biometric Enrollment and Verification, version 1.1, this applies to all different modalities. If hybrid is selected in FIA_UAU.5.1 it must be selected and the user or admin must have the option to disable the use of it.

Function 23 must be included in the ST if the function is configurable on the TOE for any of the trusted channels either mandated or selected in FTP_ITC_EXT.1.1 or FDP_UPC_EXT.1.1/APPS. The configuration can be different depending on the specific trusted channel(s) and they must be filled in for the assignment.

The assignment in function 24 consists of all externally accessible hardware ports, such as USB, the SD card, and HDMI, whose data transfer capabilities can be enabled and disabled by either the user or administrator. Disablement of data transfer over an external port must be effective during and after boot into the normal operative mode of the device. If the TOE supports states in which configured security management policy is inaccessible, for example, due to data-at-rest protection, it is acceptable to meet this requirement by ensuring that data transfer is disabled by default while in these states. Each of the ports may be enabled or disabled separately. The configuration policy need not disable all ports together. In the case of USB, charging is still allowed if data transfer capabilities have been disabled.

The assignment in function 25 consists of all protocols where the TSF acts as a server, which can be enabled and disabled by either the user or administrator.

Function 26 must be included in the selection if developer modes are supported by the TSF.

Function 27 must be included in the selection if bypass of local user authentication, such as a "Forgot Password", password hint, or remote authentication feature, is supported.

Function 29 must be included in the selection if the TSF allows applications, other than the MDM Agents, to import or remove X.509v3 certificates from the Trust Anchor Database. The MDM Agent is considered the administrator. This function does not apply to applications trusting a certificate for its own validations. The function only applies to situations where the application modifies the device-wide Trust Anchor Database, affecting the validations performed by the TSF for other applications. The user or administrator may be provided the ability to globally allow or deny any application requests in order to meet this requirement.

Function 30 must be included in the ST if "administrator is allowed to configure certificate acceptance" is selected in FIA_X509_EXT.2.2 in Functional Package for X.509, version 1.0

Function 33 should be included in the selection if FPT_TUD_EXT.5.1 is included in the ST and the configurable option is selected.

Function 34 should be included in the selection if user or administrator is selected in FCS_STG_EXT.1.4.

Function 35 should be included in the selection if the user or the administrator is selected in FCS_STG_EXT.1.5.

Function 37 must be included in the selection if FAU_SEL.1 is included in the ST.

For function 41, hotspot functionality refers to the condition in which the mobile device is serving as an access point to other devices, not the connection of the TOE to external hotspots.

Functions 42 and 43 correspond to FDP_ACF_EXT.1.2.

For function 44, FMT_SMF_EXT.2.1 specifies actions to be performed when the TOE is unenrolled from management.

Function 45 must be included in the ST if IPsec is selected in FTP_ITC_EXT.1 and the native IPsec VPN client can be configured to be Always-On. Always-On is defined as when the TOE has a network connection the VPN attempts to connect, all data leaving the device uses the VPN when the VPN is connected and no data leaves that device when the VPN is disconnected. The configuration of the VPN Client itself (with information such as VPN Gateway, certificates, and algorithms) is addressed by the PP-Module for Virtual Private Network (VPN) Clients, version 3.0.

Evaluation Activities

FMT_SMF_EXT.1
TSS
The evaluator shall verify that the TSS describes all management functions, what roles can perform each function, and how these functions are (or can be) restricted to the roles identified by FMT_MOF_EXT.1.

The following activities are organized according to the function number in the table. These activities include TSS Evaluation Activities, AGD Evaluation Activities, and test activities.

Test activities specified below shall take place in the test environment described in the evaluation activity for FPT_TUD_EXT.1.

Guidance
The evaluator shall consult the AGD guidance to perform each of the specified tests, iterating each test as necessary if both the user and administrator may perform the function. The evaluator shall verify that the AGD guidance describes how to perform each management function, including any configuration details. For each specified management function tested, the evaluator shall confirm that the underlying mechanism exhibits the configured setting.

Tests
There are no test activities for this component.

FMT_SMF_EXT.2 Specification of Remediation Actions

FMT_SMF_EXT.2.1

The TSF shall offer [**selection**: *wipe of protected data*, *wipe of sensitive data*, *remove Enterprise applications*, *remove all device-stored Enterprise resource data*, *remove Enterprise secondary authentication data*, *[**assignment**: list other available remediation actions]*] upon unenrollment and [**selection**: *[**assignment**: other administrator-configured triggers]*, *no other triggers*].

Application Note: Unenrollment may consist of removing the MDM agent or removing the administrator's policies. The functions in the selection are remediation actions that TOE may provide (perhaps via APIs) to the administrator (perhaps via an MDM agent) that may be performed upon unenrollment. "Enterprise applications" refers to applications that are in the Enterprise application group. "Enterprise resource data" refers to all stored Enterprise data and the separate resources that are available to the Enterprise application group, per FDP_ACF_EXT.2.1. If FDP_ACF_EXT.2.1 is included in the ST, then "remove all device-stored Enterprise resource data" must be selected, and is defined to be all resources selected in FDP_ACF_EXT.2.1. If FIA_UAU_EXT.4.1 is included in the ST, then "remove Enterprise secondary authentication data" must be selected. If FIA_UAU_EXT.4.1 is not included in the ST, then "remove Enterprise secondary authentication data" cannot be selected. Enterprise secondary authentication data only refers to any data stored on the

TOE that is specifically used as part of a secondary authentication mechanism to authenticate access to Enterprise applications and shared resources. Material that is used for the TOE's primary authentication mechanism or other purposes not related to authentication to or protection of Enterprise applications or shared resources should not be removed.

Protected data is all non-TSF data, including all user or enterprise data. Some or all of this data may be considered sensitive data as well. If wipe of protected data is selected it is assumed that the sensitive data is wiped as well. However, if wipe of sensitive data is selected, it does not imply that all non-TSF data (protected data) is wiped. If wipe of protected data or wipe of sensitive data is selected the wipe must be in accordance with FCS_CKM_EXT.5.1. Thus cryptographically wiping the device is an acceptable remediation action.

Evaluation Activities

FMT_SMF_EXT.2
TSS
The evaluator shall verify that the TSS describes all available remediation actions, when they are available for use, and any other administrator-configured triggers. The evaluator shall verify that the TSS describes how the remediation actions are provided to the administrator.

Guidance
There are no guidance evaluation activities for this component.

Tests
The evaluator shall use the test environment to iteratively configure the device to perform each remediation action in the selection. The evaluator shall configure the remediation action per how the TSS states it is provided to the administrator. The test environment could be an MDM agent application, but can also be an application with administrator access.

FMT_SMF_EXT.3 Current Administrator

This is an objective component.

FMT_SMF_EXT.3.1

The TSF shall provide a mechanism that allows users to view a list of currently authorized administrators and the management functions that each administrator is authorized to perform.

Evaluation Activities

FMT_SMF_EXT.3
TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests
The evaluator shall cause the TOE to be enrolled into management. The evaluator shall then invoke this mechanism and verify the ability to view that the device has been enrolled, and view the management functions that the administrator is authorized to perform.

5.1.7 Class: Protection of the TSF (FPT)

FPT_AEX_EXT.1 Application Address Space Layout Randomization

FPT_AEX_EXT.1.1

The TSF shall provide address space layout randomization ASLR to applications.

FPT_AEX_EXT.1.2

The base address of any user-space memory mapping will consist of at least 8 unpredictable bits.

Application Note: The 8 unpredictable bits may be provided by the TSF RBG (as specified in FCS_RBG.1) but is not required.

Evaluation Activities

FPT_AEX_EXT.1
TSS
The evaluator shall ensure that the TSS section of the ST describes how the 8 bits are generated and provides a justification as to why those bits are unpredictable.

Guidance
There are no guidance evaluation activities for this component.

Tests
Evaluation Activity Note: *The following test require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile Device products.*

- *Test FPT_AEX_EXT.1:1: The evaluator must select 3 apps included with the TSF. These must include any web browser or mail client included with the TSF. For each of these apps, the evaluator shall launch the same app on two separate Mobile Devices of the same type and compare all memory mapping locations. The evaluator must ensure that no memory mappings are placed in the same location on both devices.*

  If the rare (at most 1/256) chance occurs that two mappings are the same for a single app and not the same for the other two apps, the evaluator shall repeat the test with that app to verify that in the second test the mappings are different.

FPT_AEX_EXT.2 Memory Page Permissions

FPT_AEX_EXT.2.1

The TSF shall be able to enforce read, write, and execute permissions on every page of physical memory.

Evaluation Activities

FPT_AEX_EXT.2
TSS
The evaluator shall ensure that the TSS describes of the memory management unit (MMU), and ensures that this description documents the ability of the MMU to enforce read, write, and execute permissions on all pages of virtual memory.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FPT_AEX_EXT.3 Stack Overflow Protection

FPT_AEX_EXT.3.1

TSF processes that execute in a non-privileged execution domain on the application processor shall implement stack-based buffer overflow protection.

Application Note: A "non-privileged execution domain" refers to the user mode (as opposed to kernel mode, for instance) of the processor. While not all TSF processes must implement such protection, it is expected that most of the processes (to include libraries used by TSF processes) do implement buffer overflow protections.

Evaluation Activities

FPT_AEX_EXT.3
TSS
The evaluator shall determine that the TSS contains a description of stack-based buffer overflow protections implemented in the TSF software which runs in the non-privileged execution mode of the application processor. The exact implementation of stack-based buffer overflow protection will vary by platform. Example implementations may be activated through compiler options such as "-fstack-protector-all", "-fstack-protector", and "/GS" flags. The evaluator shall ensure that the TSS contains an inventory of TSF binaries and libraries, indicating those that implement stack-based buffer overflow protections as well as those that do not. The TSS must provide a rationale for those binaries and libraries that are not protected in this manner.

Guidance

There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FPT_AEX_EXT.4 Domain Isolation

FPT_AEX_EXT.4.1

The TSF shall protect itself from modification by untrusted subjects.

FPT_AEX_EXT.4.2

The TSF shall enforce isolation of address space between applications.

Application Note: In addition to the TSF software (e.g., kernel image, device drivers, trusted applications) that resides in storage, the execution context (e.g., address space, processor registers, per-process environment variables) of the software operating in a privileged mode of the processor (e.g., kernel), as well as the context of the trusted applications is to be protected. In addition to the software, any configuration information that controls or influences the behavior of the TSF is also to be protected from modification by untrusted subjects.

Configuration information includes, but is not limited to, user and administrative management function settings, WLAN profiles, and Bluetooth data such as the service-level security requirements database.

Untrusted subjects include untrusted applications; unauthorized users who have access to the device while powered off, in a screen-locked state, or when booted into auxiliary boot modes; and, unauthorized users or untrusted software or hardware which may have access to the device over a wired interface, either when the device is in a screen-locked state or booted into auxiliary boot modes.

Evaluation Activities

FPT_AEX_EXT.4
TSS
The evaluator shall ensure that the TSS describes the mechanisms that are in place that prevents non-TSF software from modifying the TSF software or TSF data that governs the behavior of the TSF. These mechanisms could range from hardware-based means (e.g. "execution rings" and memory management functionality); to software-based means (e.g. boundary checking of inputs to APIs). The evaluator determines that the described mechanisms appear reasonable to protect the TSF from modification.

The evaluator shall ensure the TSS describes how the TSF ensures that the address spaces of applications are kept separate from one another.

The evaluator shall ensure the TSS details the USSD and MMI codes available from the dialer at the locked state or during auxiliary boot modes that may alter the behavior of the TSF. The evaluator shall ensure that this description includes the code, the action performed by the TSF, and a justification that the actions performed do not modify user or TSF data. If no USSD or MMI codes are available, the evaluator shall ensure that the TSS provides a description of the method by which actions prescribed by these codes are prevented.

The evaluator shall ensure the TSS documents any TSF data (including software, execution context, configuration information, and audit logs) which may be accessed and modified over a wired interface in auxiliary boot modes. The evaluator shall ensure that the description includes data, which is modified in support of update or restore of the device. The evaluator shall ensure that this documentation includes the auxiliary boot modes in which the data may be modified, the methods for entering the auxiliary boot modes, the location of the data, the manner in which data may be modified, the data format and packaging necessary to support modification, and software or hardware tools, if any, which are necessary for modifying the data.

The evaluator shall ensure that the TSS provides a description of the means by which unauthorized and undetected modification (that is, excluding cryptographically verified updates per FPT_TUD_EXT.2) of the TSF data over the wired interface in auxiliary boots modes is prevented. The lack of publicly available tools is not sufficient justification. Examples of sufficient justification include auditing of changes, cryptographic verification in the form of a digital signature or hash, disabling the auxiliary boot modes, and access control mechanisms that prevent writing to files or flashing partitions.

Guidance
There are no guidance evaluation activities for this component.

Tests
Evaluation Activity Note: *The following tests require the vendor to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile*

Device products. In addition, the vendor provides a list of files (e.g., system files, libraries, configuration files, audit logs) that make up the TSF data. This list could be organized by folders/directories (e.g., /usr/sbin, /etc), as well as individual files that may exist outside of the identified directories.

- *Test FPT_AEX_EXT.4:1: The evaluator shall create and load an app onto the Mobile Device. This app shall attempt to traverse over all file systems and report any locations to which data can be written or overwritten. The evaluator must ensure that none of these locations are part of the OS software, device drivers, system and security configuration files, key material, or another untrusted application's image/data. For example, it is acceptable for a trusted photo editor app to have access to the data created by the camera app, but a calculator application shall not have access to the pictures.*
- *Test FPT_AEX_EXT.4:2: For each available auxiliary boot mode, the evaluator shall attempt to modify a TSF file of their choosing using the software or hardware tools described in the TSS. The evaluator shall verify that the modification fails.*

FPT_AEX_EXT.5 Kernel Address Space Layout Randomization

This is an objective component.

FPT_AEX_EXT.5.1

The TSF shall provide address space layout randomization (ASLR) to the kernel.

FPT_AEX_EXT.5.2

The base address of any kernel-space memory mapping will consist of [**assignment**: *number greater than or equal to 4*] unpredictable bits.

Application Note: The unpredictable bits may be provided by the TSF RBG (as specified in FCS_RBG.1).

Evaluation Activities

FPT_AEX_EXT.5
TSS
The evaluator shall ensure that the TSS section of the ST describes how the bits are generated and provides a justification as to why those bits are unpredictable.

Guidance
There are no guidance evaluation activities for this component.

Tests
Evaluation Activity Note: *The following test require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile Device products.*

- *Test FPT_AEX_EXT.5:1: The evaluator shall reboot the TOE six times. For each of these reboots, the evaluator shall examine memory mapping locations of the kernel. The evaluator must ensure that for at least five reboots the memory mappings are not placed in the same location on both devices.*

FPT_AEX_EXT.6 Write or Execute Memory Page Permissions

This is an objective component.

FPT_AEX_EXT.6.1

The TSF shall prevent write and execute permissions from being simultaneously granted to any page of physical memory [**selection**: *with no exceptions*, *[**assignment**: specific exceptions]*].

Application Note: Memory used for just-in-time (JIT) compilation is anticipated as an exception in this requirement; if so, the ST author must address how this exception is permitted. It is expected that the memory management unit will transition the system to a non-operational state if any violation is detected in kernel memory space.

Evaluation Activities

FPT_AEX_EXT.6
TSS
The evaluator shall ensure that the TSS describes how the operating system of the application

processor prevents all processes executing in a non-privileged execution domain from achieving write and execute permissions on any page of memory (with only specified exceptions). The evaluator shall ensure that the TSS describes how such processes are unable to request pages of memory with such permissions, and how they are unable to change permissions to both write and execute on any pages already allocated to them.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FPT_AEX_EXT.7 Heap Overflow Protection

This is an objective component.

FPT_AEX_EXT.7.1

The TSF shall include heap-based buffer overflow protections in the runtime environment it provides to processes that execute on the application processor.

Application Note: These heap-based buffer overflow protections are expected to ensure the integrity of heap metadata such as memory addresses or offsets recorded by the heap implementation to manage memory blocks. This includes chunk headers, look-aside lists, and other data structures used to track the state and location of memory blocks managed by the heap.

Evaluation Activities

FPT_AEX_EXT.7
TSS
The evaluator shall verify that the TSS enumerates the heap implementations provided to userspace processes. The evaluator shall ensure that the TSS lists all types of heap metadata and identifies how the integrity of each type of metadata is ensured. The evaluator shall ensure that the TSS identifies all fields within each type of metadata and identifies how the integrity of these fields is ensured. The evaluator shall verify that the TSS identifies the manner in which an error condition is entered when a heap overflow is detected and the resulting actions taken by the TSF.

Guidance
There are no guidance evaluation activities for this component.

Tests
For each heap implementation, the evaluator shall write, or the developer shall provide access to, an application, which allocates memory from the heap and then writes arbitrary data significantly beyond the end of the allocated buffer. The evaluator shall attempt to execute this application and verify that the write is not allowed.

FPT_BBD_EXT.1 Application Processor Mediation

This is an objective component.

FPT_BBD_EXT.1.1

The TSF shall prevent code executing on any baseband processor (BP) from accessing application processor (AP) resources except when mediated by the AP.

Application Note: These resources include:

- Volatile and non-volatile memory
- Control of and data from integrated and non-integrated peripherals (e.g. USB controllers, touch screen controllers, LCD controller, codecs)
- Control of and data from integrated and non-integrated I/O sensors (e.g. camera, light, microphone, GPS, accelerometers, geomagnetic field sensors)

Mobile devices are becoming increasingly complex having an application processor that runs an operating system and user applications and separate baseband processors that handle cellular and other wireless network connectivity.

- The application processor within most modern Mobile Devices is a system

on a chip (SoC) that integrates, for example, CPU/GPU cores and memory interface electronics into a single, power-efficient package.
- Baseband processors are becoming increasingly complex themselves delivering voice encoding alongside multiple independent radios (LTE, Wi-Fi, Bluetooth, FM, GPS) in a single package containing multiple CPUs and DSPs.

Thus, the baseband processors in these requirements include such integrated SoCs and include any radio processors (integrated or not) on the Mobile Device.

All other requirements mostly, except where noted, apply to firmware/software on the application processor, but future requirements (notably, all Integrity, Access Control, and Anti-Exploitation requirements) will apply to application processors and baseband processors.

Evaluation Activities

FPT_BBD_EXT.1
TSS
The evaluator shall ensure that the TSS section of the ST describes at a high level how the processors on the Mobile Device interact, including which bus protocols they use to communicate, any other devices operating on that bus (peripherals and sensors), and identification of any shared resources. The evaluator shall verify that the design described in the TSS does not permit any BPs from accessing any of the peripherals and sensors or from accessing main memory (volatile and non-volatile) used by the AP. In particular, the evaluator shall ensure that the design prevents modification of executable memory of the AP by the BP.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FPT_BLT_EXT.1 Limitation of Bluetooth Profile Support

This is an objective component.

FPT_BLT_EXT.1.1

The TSF shall disable support for [**assignment**: *list of Bluetooth profiles*] Bluetooth profiles when they are not currently being used by an application on the Mobile Device, and shall require explicit user action to enable them.

Application Note: Some Bluetooth services incur more serious consequences if unauthorized remote devices gain access to them. Such services should be protected by measures like disabling support for the associated Bluetooth profile unless it is actively being used by an application on the Mobile Device (in order to prevent discovery by a Service Discovery Protocol search), and then requiring explicit user action to enable those profiles in order to use the services. It may be further appropriate to require additional user action before granting a remote device access to that service.

For example, it may be appropriate to disable the OBEX Push Profile until a user on the Mobile Device pushes a button in an application indicating readiness to transfer an object. After completion of the object transfer, support for the OBEX profile should be suspended until the next time the user requests its use.

Evaluation Activities

FPT_BLT_EXT.1
TSS
The evaluator shall ensure that the TSS lists all Bluetooth profiles that are disabled while not in use by an application and which need explicit user action in order to become enabled.

Guidance
There are no guidance evaluation activities for this component.

Tests
The evaluator shall perform the following tests:

- *Test FPT_BLT_EXT.1:1: While the service is not in active use by an application on the TOE,*

the evaluator shall attempt to discover a service associated with a "protected" Bluetooth profile (as specified by the requirement) on the TOE via a Service Discovery Protocol search. The evaluator shall verify that the service does not appear in the Service Discovery Protocol search results. Next, the evaluator shall attempt to gain remote access to the service from a device that does not currently have a trusted device relationship with the TOE. The evaluator shall verify that this attempt fails due to the unavailability of the service and profile.

- *Test FPT_BLT_EXT.1:2: The evaluator shall repeat Test 1 with a device that currently has a trusted device relationship with the TOE and verify that the same behavior is exhibited.*

FPT_FLS.1 Failure with Preservation of Secure State

FPT_FLS.1.1

The TSF shall preserve a secure state when the following types of failures occur: [*DRBG self-test failure*].

Application Note: The intent of this requirement is to ensure that cryptographic services requiring random bit generation cannot be performed if a failure of a self-test defined in FPT_TST.1 occurs.

Evaluation Activities

FPT_FLS.1
TSS
The evaluator shall verify that the TSF describes how the TOE enters an error state in the event of a DRBG self-test failure.

Guidance
The evaluator shall verify that the guidance documentation describes the error state that results from a DRBG self-test failure and the actions that a user or administrator should take in response to attempt to resolve the error state.

Tests
There are no test activities for this component.

FPT_JTA_EXT.1 JTAG Disablement

FPT_JTA_EXT.1.1

The TSF shall [**selection**: *disable access through hardware, control access by a signing key*] to JTAG.

Application Note: This requirement means that access to JTAG must be disabled either through hardware or restricted through the use of a signing key.

Evaluation Activities

FPT_JTA_EXT.1
TSS
If disable access through hardware is selected:
The evaluator shall examine the TSS to determine the location of the JTAG ports on the TSF, to include the order of the ports (i.e. Data In, Data Out, Clock, etc.).

If control access by a signing key is selected:
The evaluator shall examine the TSS to determine how access to the JTAG is controlled by a signing key. The evaluator shall examine the TSS to determine when the JTAG can be accessed, i.e. what has the access to the signing key.

Guidance
There are no guidance evaluation activities for this component.

Tests
Evaluation Activity Note:*The following test requires the developer to provide access to a test platform that provides the evaluator with chip level access.*

If disable access through hardware is selected:
The evaluator shall connect a packet analyzer to the JTAG ports. The evaluator shall query the JTAG port for its device ID and confirm that the device ID cannot be retrieved.

FPT_KST_EXT.1 Key Storage

FPT_KST_EXT.1.1

The TSF shall not store any plaintext key material in readable non-volatile memory.

Application Note: The intention of this requirement is that the TOE will not write plaintext keying material to persistent storage. For the purposes of this requirement, keying material refers to authentication data, passwords, secret/private symmetric keys, private asymmetric keys, data used to derive keys, etc. These values must be stored encrypted.

This requirement also applies to any value derived from passwords. Thus, the TOE cannot store plaintext password hashes for comparison purposes before protected data is decrypted, and the TOE should use key derivation and decryption to verify the Password Authentication Factor.

If hybrid is selected in FIA_UAU.5.1 keying material also refers to the PIN/password used as part of the hybrid authentication.

Evaluation Activities

FPT_KST_EXT.1
TSS
The evaluator shall consult the TSS section of the ST in performing the Evaluation Activities for this requirement.

In performing their review, the evaluator shall determine that the TSS contains a description of the activities that happen on power-up and password authentication relating to the decryption of DEKs, stored keys, and data.

The evaluator shall ensure that the description also covers how the cryptographic functions in the FCS requirements are being used to perform the encryption functions, including how the KEKs, DEKs, and stored keys are unwrapped, saved, and used by the TOE so as to prevent plaintext from being written to non-volatile storage. The evaluator shall ensure that the TSS describes, for each power-down scenario how the TOE ensures that all keys in non-volatile storage are not stored in plaintext.

The evaluator shall ensure that the TSS describes how other functions available in the system (e.g., regeneration of the keys) ensure that no unencrypted key material is present in persistent storage.

The evaluator shall review the TSS to determine that it makes a case that key material is not written unencrypted to the persistent storage.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FPT_KST_EXT.2 No Key Transmission

FPT_KST_EXT.2.1

The TSF shall not transmit any plaintext key material outside the security boundary of the TOE.

Application Note: The intention of this requirement is to prevent the logging of plaintext key information to a service that transmits the information off-device. For the purposes of this requirement, key material refers to keys, passwords, and other material that is used to derive keys.

If hybrid is selected in FIA_UAU.5.1 keying material also refers to the PIN/password used as part of the hybrid authentication.

In the future, this requirement will apply to symmetric and asymmetric private keys stored in the TOE secure key storage where applications are outside the boundary of the TOE. Thus, the TSF will be required to provide cryptographic key operations (signature, encryption, and decryption) on behalf of applications (FCS_SRV_EXT.2.1) that have access to those keys.

Evaluation Activities

FPT_KST_EXT.2
TSS
The evaluator shall consult the TSS section of the ST in performing the Evaluation Activities for this requirement. The evaluator shall ensure that the TSS describes the TOE security boundary. The cryptographic module may very well be a particular kernel module, the Operating System,

the Application Processor, or up to the entire Mobile Device.

In performing their review, the evaluator shall determine that the TSS contains a description of the activities that happen on power-up and password authentication relating to the decryption of DEKs, stored keys, and data.

The evaluator shall ensure that the TSS describes how other functions available in the system (e.g., regeneration of the keys) ensure that no unencrypted key material is transmitted outside the security boundary of the TOE.

The evaluator shall review the TSS to determine that it makes a case that key material is not transmitted outside the security boundary of the TOE.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FPT_KST_EXT.3 No Plaintext Key Export

FPT_KST_EXT.3.1

The TSF shall ensure it is not possible for the TOE users to export plaintext keys.

Application Note: Plaintext keys include DEKs, KEKs, and all keys stored in the secure key storage (FCS_STG_EXT.1). The intent of this requirement is to prevent the plaintext keys from being exported during a backup authorized by the TOE user or administrator.

Evaluation Activities

FPT_KST_EXT.3
TSS
The ST author will provide a statement of their policy for handling and protecting keys. The evaluator shall check to ensure the TSS describes a policy in line with not exporting either plaintext DEKs, KEKs, or keys stored in the secure key storage.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FPT_NOT_EXT.1 Self-Test Notification

FPT_NOT_EXT.1.1

The TSF shall transition to non-operational mode and [**selection**: *log failures in the audit record, notify the administrator, [**assignment**: other actions], no other actions*] when the following types of failures occur:

- failures of the self-tests
- TSF software integrity verification failures
- [**selection**: *no other failures, [**assignment**: other failures]*]

Evaluation Activities

FPT_NOT_EXT.1
TSS
The evaluator shall verify that the TSS describes critical failures that may occur and the actions to be taken upon these critical failures.

Guidance
There are no guidance evaluation activities for this component.

Tests
Evaluation Activity Note: *The following test require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile Device products.*

- *Test FPT_NOT_EXT.1:1: The evaluator shall use a tool provided by the developer to modify*

files and processes in the system that correspond to critical failures specified in the second list. The evaluator shall verify that creating these critical failures causes the device to take the remediation actions specified in the first list.

FPT_NOT_EXT.2 Software Integrity Verification

This is an objective component.

FPT_NOT_EXT.2.1

The TSF shall [**selection**: *audit, provide the administrator with*] TSF-software integrity verification values.

Application Note: These notifications are typically called remote attestation and these integrity values are typically called measurements. The integrity values are calculated from hashes of critical memory and values, including executable code. The ST author must select whether these values are logged as a part of FAU_GEN.1.1 or are provided to the administrator.

FPT_NOT_EXT.2.2

The TSF shall cryptographically sign all integrity verification values.

Application Note: The intent of this requirement is to provide assurance to the administrator that the responses provided are from the TOE and have not been modified or spoofed by a man-in-the-middle such as a network-based adversary or a malicious MDM Agent.

Evaluation Activities

FPT_NOT_EXT.2.1
TSS
The evaluator shall verify that the TSS describes which critical memory is measured for these integrity values and how the measurement is performed (including which TOE software performs these generates these values, how that software accesses the critical memory, and which algorithms are used).

Guidance
If the integrity values are provided to the administrator, the evaluator shall verify that the AGD guidance contains instructions for retrieving these values and information for interpreting them. For example, if multiple measurements are taken, what those measurements are and how changes to those values relate to changes in the device state.

Tests
Evaluation Activity Note: *The following test may require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile Device products.*

The evaluator shall repeat the following test for each measurement:

- *Test FPT_NOT_EXT.2.1:1: The evaluator shall boot the device in an approved state and record the measurement taken (either from the log or by using the administrative guidance to retrieve the value via an MDM Agent). The evaluator shall modify the critical memory or value that is measured. The evaluator shall boot the device and verify that the measurement changed.*

FPT_NOT_EXT.2.2
TSS
The evaluator shall verify that the TSS describes which key the TSF uses to sign the responses to queries and the certificate used to prove ownership of the key, and the method of associating the certificate with a particular device manufacturer and model.

Guidance
There are no guidance evaluation activities for this component.

Tests
The evaluator shall perform the following test:

- *Test FPT_NOT_EXT.2.2:1: The evaluator shall write, or the developer shall provide, a management application that queries either the audit logs or the measurements. The evaluator shall verify that the responses to these queries are signed and verify the signatures against the TOE's certificate.*

FPT_STM.1 Reliable Time Stamps

FPT_STM.1.1

The TSF shall be able to provide reliable time stamps **for its own use**.

Evaluation Activities

FPT_STM.1
TSS
The evaluator shall examine the TSS to ensure that it lists each security function that makes use of time. The TSS provides a description of how the time is maintained and considered reliable in the context of each of the time related functions. This documentation must identify whether the TSF uses a NTP server or the carrier's network time as the primary time sources.

Guidance
The evaluator examines the operational guidance to ensure it describes how to set the time.

Tests
- *Test FPT_STM.1:1: The evaluator uses the operational guide to set the time. The evaluator shall then use an available interface to observe that the time was set correctly.*

FPT_TST.1 TSF Self-Testing

FPT_TST.1.1

The TSF shall run a suite of the following self-tests [**selection**: *during initial start-up, periodically during normal operation, at the request of the authorized user, at the conditions [**assignment**: conditions under which self-test should occur]*] to demonstrate the correct operation of [*TSF DRBG specified in FCS_RBG.1*].

FPT_TST.1.2

The TSF shall provide authorized users with the capability to verify the integrity of [*[DRBG seed/output data]*].

FPT_TST.1.3

The TSF shall provide authorized users with the capability to verify the integrity of [*[TSF DRBG specified in FCS_RBG.1]*].

Application Note: This SFR is a required dependency of FCS_RBG.1. It is intended to require that any DRBG implemented by the TOE undergo health testing to ensure that the random bit generation functionality has not been degraded. If the TSF supports multiple DRBGs, this SFR should be iterated to describe the self-test behavior for each.

Evaluation Activities

FPT_TST.1
TSS
The evaluator shall examine the TSS to ensure that it details the self-tests that are run by the TSF along with how they are run. This description should include an outline of what the tests are actually doing. The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the DRBG is operating correctly.

Note that this information may also be placed in the entropy documentation specified by Appendix D - Entropy Documentation And Assessment.

Guidance
If a self-test can be executed at the request of an authorized user, the evaluator shall verify that the operational guidance provides instructions on how to execute that self-test.

Tests
For each self-test, the evaluator shall verify that evidence is produced that the self-test is executed when specified by FPT_TST.1.1.

If a self-test can be executed at the request of an authorized user, the evaluator shall verify that following the steps documented in the operational guidance to perform the self-test will result in execution of the self-test.

FPT_TST_EXT.1 TSF Cryptographic Functionality Testing

FPT_TST_EXT.1.1

The TSF shall run a suite of self-tests during initial start-up (on power on) to demonstrate the correct operation of all cryptographic functionality.

Application Note: This requirement may be met by performing known answer tests or pair-wise consistency tests. The self-tests must be performed before the cryptographic functionality is exercised (for example, during the initialization of

a process that utilizes the functionality).

The cryptographic functionality includes the cryptographic operations in FCS_COP, the key generation functions in FCS_CKM, and the random bit generation in FCS_RBG.1.

Evaluation Activities

FPT_TST_EXT.1
TSS
The evaluator shall examine the TSS to ensure that it specifies the self-tests that are performed at start-up. This description must include an outline of the test procedures conducted by the TSF (e.g., rather than saying "memory is tested", a description similar to "memory is tested by writing a value to each memory location and reading it back to ensure it is identical to what was written" shall be used). The TSS must include any error states that they TSF may enter when self-tests fail, and the conditions and actions necessary to exit the error states and resume normal operation. The evaluator shall verify that the TSS indicates these self-tests are run at start-up automatically, and do not involve any inputs from or actions by the user or operator.

The evaluator shall inspect the list of self-tests in the TSS and verify that it includes algorithm self-tests. The algorithm self-tests will typically be conducted using known answer tests.

Guidance
There are no guidance evaluation activities for this component.

Tests
There are no test evaluation activities for this component.

FPT_TST_EXT.2/POSTKERNEL TSF Integrity Checking (Post-Kernel)

This is an objective component.

FPT_TST_EXT.2.1/POSTKERNEL
The TSF shall verify the integrity of [**selection**: *all executable code*, *[**assignment**: subset of executable code]*] stored in mutable media prior to its execution through the use of [**selection**: *a digital signature using an immutable hardware asymmetric key, an immutable hardware hash of an asymmetric key, an immutable hardware hash, a digital signature using a hardware-protected asymmetric key, hardware-protected hash*].

Application Note: All executable code covered in this requirement is executed after the kernel is loaded.

If "all executable code in mutable media" is verified, implementation in hardware or in read-only memory is a natural logical consequence.

At this time, the verification of software executed on other processors stored in mutable media is not required; however, it may be added in the first assignment. If all executable code (including bootloaders, kernel, device drivers, pre-loaded applications, user-loaded applications, and libraries) is verified, "all executable code stored in mutable media" should be selected.

Evaluation Activities

FPT_TST_EXT.2/POSTKERNEL
TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests
The evaluation activity shall be completed in conjunction with FPT_TST_EXT.2/PREKERNEL for all executable code specified.

FPT_TST_EXT.2/PREKERNEL TSF Integrity Checking (Pre-Kernel)

FPT_TST_EXT.2.1/PREKERNEL
The TSF shall verify the integrity of [*the bootchain up through the Application Processor OS kernel*] stored in mutable media prior to its execution through the

use of [**selection**: *a digital signature using an immutable hardware asymmetric key, an immutable hardware hash of an asymmetric key, an immutable hardware hash, a digital signature using a hardware-protected asymmetric key*].

Application Note: The bootchain of the TSF is the sequence of firmware and software, including ROM, bootloaders, and kernel, which ultimately result in loading the kernel on the Application Processor, regardless of which processor executes that code. Executable code that would be loaded after the kernel is covered in FPT_TST_EXT.2/POSTKERNEL.

In order to meet this requirement, the hardware protection may be transitive in nature: a hardware-protected public key, hash of an asymmetric key, or hash may be used to verify the mutable bootloader code which contains a key or hash used by the bootloader to verify the mutable OS kernel code, which contains a key or hash to verify the next layer of executable code, and so on.

The cryptographic mechanism used to verify the (initial) mutable executable code must be protected, such as being implemented in hardware or in read-only memory (ROM).

Evaluation Activities

FPT_TST_EXT.2/PREKERNEL
TSS
The evaluator shall verify that the TSS section of the ST includes a description of the boot procedures, including a description of the entire bootchain, of the software for the TSF's Application Processor. The evaluator shall ensure that before loading the bootloaders for the operating system and the kernel, all bootloaders and the kernel software itself is cryptographically verified. For each additional category of executable code verified before execution, the evaluator shall verify that the description in the TSS describes how that software is cryptographically verified.

The evaluator shall verify that the TSS contains a justification for the protection of the cryptographic key or hash, preventing it from being modified by unverified or unauthenticated software. The evaluator shall verify that the TSS contains a description of the protection afforded to the mechanism performing the cryptographic verification.

The evaluator shall verify that the TSS describes each auxiliary boot mode available on the TOE during the boot procedures. The evaluator shall verify that, for each auxiliary boot mode, a description of the cryptographic integrity of the executed code through the kernel is verified before each execution.

Guidance
There are no guidance evaluation activities for this component.

Tests
Evaluation Activity Note: *The following tests require the vendor to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile Device products.*

The evaluator shall perform the following tests:

- *Test FPT_TST_EXT.2/PREKERNEL:1: The evaluator shall perform actions to cause TSF software to load and observe that the integrity mechanism does not flag any executables as containing integrity errors and that the TOE properly boots.*
- *Test FPT_TST_EXT.2/PREKERNEL:2: The evaluator shall modify a TSF executable that is integrity protected and cause that executable to be successfully loaded by the TSF. The evaluator observes that an integrity violation is triggered and the TOE does not boot. (Care must be taken so that the integrity violation is determined to be the cause of the failure to load the module, and not the fact that the module was modified so that it was rendered unable to run because its format was corrupt).*
- *Test FPT_TST_EXT.2/PREKERNEL:3: [conditional] If the ST author indicates that the integrity verification is performed using a public key, the evaluator shall verify that the update mechanism includes a certificate validation according to FIA_X509_EXT.1 as defined in Functional Package for X.509, version 1.0. The evaluator shall digitally sign the TSF executable with a certificate that does not have the Code Signing purpose in the extendedKeyUsage field and verify that an integrity violation is triggered. The evaluator shall repeat the test using a certificate that contains the Code Signing purpose and verify that the integrity verification succeeds. Ideally, the two certificates should be identical except for the extendedKeyUsage field.*

FPT_TST_EXT.3 TSF Integrity Testing

This is a selection-based component. Its inclusion depends upon selection from .

FPT_TST_EXT.3.1

The TSF shall not execute code if the code signing certificate is deemed invalid.

Application Note: Certificates may optionally be used for code signing for integrity verification (FPT_TST_EXT.2.1/PREKERNEL). If "code signing for software integrity testing" is selected in FIA_X509_EXT.2.1 in Functional Package for X.509, version 1.0, FPT_TST_EXT.3.1 must be included in the ST.

Validity is determined by the certificate path, the expiration date, and the revocation status in accordance with RFC 5280.

Evaluation Activities

FPT_TST_EXT.3
TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests
Testing for this component is performed in conjunction with the Evaluation Activities for FPT_TST_EXT.2.1/PREKERNEL.

FPT_TUD_EXT.1 TSF Version Query

FPT_TUD_EXT.1.1

The TSF shall provide authorized users the ability to query the current version of the TOE firmware/software.

FPT_TUD_EXT.1.2

The TSF shall provide authorized users the ability to query the current version of the hardware model of the device.

Application Note: The current version of the hardware model of the device is an identifier that is sufficient to indicate (in tandem with manufacturer documentation) the hardware which comprises the device.

FPT_TUD_EXT.1.3

The TSF shall provide authorized users the ability to query the current version of installed mobile applications.

Application Note: The current version of mobile applications is the name and published version number of each installed mobile application.

Evaluation Activities

FPT_TUD_EXT.1
The evaluator shall establish a test environment consisting of the Mobile Device and any supporting software that demonstrates usage of the management functions. This can be test software from the developer, a reference implementation of management software from the developer, or other commercially available software. The evaluator shall set up the Mobile Device and the other software to exercise the management functions according to the provided guidance documentation.

TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests

- *Test FPT_TUD_EXT.1:1: Using the AGD guidance provided, the evaluator shall test that the administrator and user can query:*
  - *The current version of the TSF operating system and any firmware that can be updated separately*
  - *The hardware model of the TSF*
  - *The current version of all installed mobile applications*

The evaluator must review manufacturer documentation to ensure that the hardware model identifier is sufficient to identify the hardware which comprises the device.

FPT_TUD_EXT.2 TSF Update Verification

FPT_TUD_EXT.2.1

The TSF shall verify software updates to the Application Processor system software and [**selection**: ***[assignment**: other processor system software] , no other processor system software*] using a digital signature verified by the manufacturer trusted key prior to installing those updates.

Application Note: The digital signature mechanism is implemented in accordance with FCS_COP.1.1/SIGN.

At this time, this requirement does not require verification of software updates to the software operating outside the Application Processor.

Any change, via a supported mechanism, to software residing in non-volatile storage is deemed a software update. Thus, this requirement applies to TSF software updates regardless of how the software arrives or is delivered to the device. This includes over-the-air (OTA) updates as well as partition images containing software which may be delivered to the device over a wired interface.

FPT_TUD_EXT.2.2

The TSF shall [**selection**: *never update*, *update only by verified software*] the TSF boot integrity [**selection**: *key*, *hash*].

Application Note: The key or hash updated via this requirement is used for verifying software before execution in FPT_TST_EXT.2/PREKERNEL. The key or hash is verified as a part of the digital signature on an update, and the software which performs the update of the key or hash is verified by FPT_TST_EXT.2/PREKERNEL.

FPT_TUD_EXT.2.3

The TSF shall verify that the digital signature verification key used for TSF updates [**selection**: *is validated to a public key in the Trust Anchor Database*, *matches an immutable hardware public key*].

Application Note: The ST author must indicate the method by which the signing key for system software updates is limited and, if selected in FPT_TUD_EXT.2.3, must indicate how this signing key is protected by the hardware.

If certificates are used, certificates are validated for the purpose of software updates in accordance with FIA_X509_EXT.1 in Functional Package for X.509, version 1.0 and should be selected in FIA_X509_EXT.2.1 in Functional Package for X.509, version 1.0. Additionally, FPT_TUD_EXT.4 must be included in the ST.

Evaluation Activities

FPT_TUD_EXT.2
TSS
The evaluator shall verify that the TSS section of the ST describes all TSF software update mechanisms for updating the system software. The evaluator shall verify that the description includes a digital signature verification of the software before installation and that installation fails if the verification fails. The evaluator shall verify that all software and firmware involved in updating the TSF is described and, if multiple stages and software are indicated, that the software/firmware responsible for each stage is indicated and that the stages which perform signature verification of the update are identified.

The evaluator shall verify that the TSS describes the method by which the digital signature is verified and that the public key used to verify the signature is either hardware-protected or is validated to chain to a public key in the Trust Anchor Database. If hardware-protection is selected, the evaluator shall verify that the method of hardware-protection is described and that the ST author has justified why the public key may not be modified by unauthorized parties.

[conditional] If the ST author indicates that software updates to system software running on other processors is verified, the evaluator shall verify that these other processors are listed in the TSS and that the description includes the software update mechanism for these processors, if different than the update mechanism for the software executing on the Application Processor.

[conditional] If the ST author indicates that the public key is used for software update digital signature verification, the evaluator shall verify that the update mechanism includes a certificate validation according to FIA_X509_EXT.1 and a check for the Code Signing purpose in the extendedKeyUsage.

Guidance
There are no guidance evaluation activities for this component.

Tests
The evaluator shall verify that the developer has provided evidence that the following tests were performed for each available update mechanism:

- *Test FPT_TUD_EXT.2:1: The tester shall try to install an update without the digital signature and shall verify that installation fails. The tester shall attempt to install an update with digital signature, and verify that installation succeeds.*
- *Test FPT_TUD_EXT.2:2: The tester shall digitally sign the update with a key disallowed by the device and verify that installation fails. The tester shall attempt to install an update signed with the allowed key and verify that installation succeeds.*
- *Test FPT_TUD_EXT.2:3: [conditional] The tester shall digitally sign the update with an invalid certificate and verify that update installation fails. The tester attempt to install an update that was digitally signed using a valid certificate and a certificate that contains the purpose and verify that the update installation succeeds.*
- *Test FPT_TUD_EXT.2:4: [conditional] The tester shall repeat these test for the software executing on each processor listed in the first selection. The tester shall attempt to install an update without the digital signature and shall verify that installation fails. The tester shall attempt to install an update with digital signature, and verify that installation succeeds.*

FPT_TUD_EXT.3 Application Signing

FPT_TUD_EXT.3.1

The TSF shall verify mobile application software using a digital signature mechanism prior to installation.

Application Note: This requirement does not necessitate an X.509v3 certificate or certificate validation. X.509v3 certificates and certificate validation are addressed in FPT_TUD_EXT.5.1.

Evaluation Activities

FPT_TUD_EXT.3
TSS
The evaluator shall verify that the TSS describes how mobile application software is verified at installation. The evaluator shall ensure that this method uses a digital signature.

Guidance
There are no guidance evaluation activities for this component.

Tests
Evaluation Activity Note: *The following test does not have to be tested using the commercial application store.*

- *Test FPT_TUD_EXT.3:1: The evaluator shall write, or the developer shall provide access to, an application. The evaluator shall try to install this application without a digitally signature and shall verify that installation fails. The evaluator shall attempt to install a digitally signed application, and verify that installation succeeds.*

FPT_TUD_EXT.4 Trusted Update Verification

This is a selection-based component. Its inclusion depends upon selection from .

FPT_TUD_EXT.4.1

The TSF shall not install code if the code signing certificate is deemed invalid.

Application Note: Certificates may optionally be used for code signing of system software updates (FPT_TUD_EXT.2.3) and of mobile applications (FPT_TUD_EXT.5.1). This element must be included in the ST if certificates are used for either update element. If "code signing for system software updates" is selected or the assignment is selected and specifies the use of X.509 in code signing for mobile applications in FIA_X509_EXT.2.1 in Functional Package for X.509, version 1.0, FPT_TUD_EXT.4 must be included in the ST.

Validity is determined by the certificate path, the expiration date, and the revocation status in accordance with RFC 5280.

Evaluation Activities

FPT_TUD_EXT.4
TSS
There are no TSS evaluation activities for this component.

Guidance
There are no guidance evaluation activities for this component.

Tests
Testing for this component is performed in conjunction with the Evaluation Activities for FPT_TUD_EXT.2 and FPT_TUD_EXT.5.

FPT_TUD_EXT.5 Application Verification

This is an objective component.

FPT_TUD_EXT.5.1

The TSF shall by default only install mobile applications cryptographically verified by [**selection**: *a built-in X.509v3 certificate*, *a configured X.509v3 certificate*].

Application Note: The built-in certificate is installed by the manufacturer either at time of manufacture or as a part of system updates. The configured certificate used to verify the signature is set according to FMT_SMF_EXT.1 function 33.

Evaluation Activities

FPT_TUD_EXT.5
TSS
The evaluator shall verify that the TSS describes how mobile application software is verified at installation. The evaluator shall ensure that this method uses a digital signature by a code signing certificate.

Guidance
There are no guidance evaluation activities for this component.

Tests

- *Test FPT_TUD_EXT.5:1: The evaluator shall write, or the developer shall provide access to, an application. The evaluator shall try to install this application without a digitally signature and shall verify that installation fails. The evaluator shall attempt to install an application digitally signed with an appropriate certificate, and verify that installation succeeds.*
- *Test FPT_TUD_EXT.5:2: The evaluator shall digitally sign the application with an invalid certificate and verify that application installation fails. The evaluator shall digitally sign the application with a certificate that does not have the Code Signing purpose and verify that application installation fails. This test may be performed in conjunction with the Evaluation Activities for FIA_X509_EXT.1 as defined in Functional Package for X.509, version 1.0.*
- *Test FPT_TUD_EXT.5:3: If necessary, the evaluator shall configure the device to limit the public keys that can sign application software according to the AGD guidance. The evaluator shall digitally sign the application with a certificate disallowed by the device or configuration and verify that application installation fails. The evaluator shall attempt to install an application digitally signed with an authorized certificate and verify that application installation succeeds.*

FPT_TUD_EXT.6 Trusted Update Verification

This is an objective component.

FPT_TUD_EXT.6.1

The TSF shall verify that software updates to the TSF are a current or later version than the current version of the TSF.

Application Note: A later version has a larger version number. The method for distinguishing newer software versions from older versions is determined by the manufacturer.

Evaluation Activities

FPT_TUD_EXT.6
TSS
The evaluator shall verify that the TSS describes the mechanism that prevents the TSF from installing software updates that are an older version that the currently installed version.

Guidance
There are no guidance evaluation activities for this component.

Tests
The evaluator shall repeat the following tests to cover all allowed software update mechanisms as described in the TSS. For example, if the update mechanism replaces an entire partition containing many separate code files, the evaluator does not need to repeat the test for each individual file.

- *Test FPT_TUD_EXT.6:1: The evaluator shall attempt to install an earlier version of software (as determined by the manufacturer). The evaluator shall verify that this attempt fails by checking the version identifiers or cryptographic hashes of the privileged software against those previously recorded and checking that the values have not changed.*
- *Test FPT_TUD_EXT.6:2: The evaluator shall attempt to install a current or later version and shall verify that the update succeeds.*

5.1.8 Class: TOE Access (FTA)

FTA_SSL_EXT.1 TSF- and User-initiated Locked State

FTA_SSL_EXT.1.1

The TSF shall transition to a locked state after a time interval of inactivity.

FTA_SSL_EXT.1.2

The TSF shall transition to a locked state after initiation by either the user or the administrator.

FTA_SSL_EXT.1.3

The TSF shall, upon transitioning to the locked state, perform the following operations:

- Clearing or overwriting display devices, obscuring the previous contents;
- [**assignment**: *Other actions performed upon transitioning to the locked state*].

Application Note: The time interval of inactivity is configured using FMT_SMF_EXT.1 function 2. The user/administrator-initiated lock is specified in FMT_SMF_EXT.1 function 6.

Evaluation Activities

FTA_SSL_EXT.1
TSS
The evaluator shall verify the TSS describes the actions performed upon transitioning to the locked state.

Guidance
The evaluation shall verify that the AGD guidance describes the method of setting the inactivity interval and of commanding a lock. The evaluator shall verify that the TSS describes the information allowed to be displayed to unauthorized users.

Tests

- *Test FTA_SSL_EXT.1:1: The evaluator shall configure the TSF to transition to the locked state after a time of inactivity (FMT_SMF_EXT.1) according to the AGD guidance. The evaluator shall wait until the TSF locks and verify that the display is cleared or overwritten and that the only actions allowed in the locked state are unlocking the session and those actions specified in FIA_UAU_EXT.2.*
- *Test FTA_SSL_EXT.1:2: The evaluator shall command the TSF to transition to the locked state according to the AGD guidance as both the user and the administrator. The evaluator shall wait until the TSF locks and verify that the display is cleared or overwritten and that the only actions allowed in the locked state are unlocking the session and those actions specified in FIA_UAU_EXT.2.*

FTA_TAB.1 Default TOE Access Banners

FTA_TAB.1.1

Before establishing a user session, the [*TSF*] shall display an [*advisory warning*] message **regarding unauthorized use of the TOE**.

Application Note: This requirement may be met with the configuration of either text or an image containing the text of the desired message. The TSF must minimally display this information at startup, but may also display the information at every unlock. The banner is configured according to FMT_SMF_EXT.1 function 36.

Evaluation Activities

FTA_TAB.1
TSS
The TSS shall describe when the banner is displayed.

Guidance
There are no guidance evaluation activities for this component.

Tests
The evaluator shall also perform the following test:

- *Test FTA_TAB.1:1: The evaluator follows the operational guidance to configure a notice and consent warning message. The evaluator shall then start up or unlock the TSF. The evaluator shall verify that the notice and consent warning message is displayed in each instance described in the TSS.*

5.1.9 Class: Trusted Path/Channels (FTP)

FTP_ITC_EXT.1 Trusted Channel Communication

FTP_ITC_EXT.1.1

The TSF shall use

- 802.11-2012 in accordance with the [*PP-Module for Wireless LAN Clients, version 2.0*],
- 802.1X in accordance with the [*PP-Module for Wireless LAN Clients, version 2.0*],
- EAP-TLS in accordance with the [*PP-Module for Wireless LAN Clients, version 2.0*],
- Mutually authenticated TLS in accordance with the [*Functional Package for Transport Layer Security (TLS), version 2.1*]

and [**selection**:

- *IPsec in accordance with the PP-Module for Virtual Private Network (VPN) Clients, version 3.0*
- *mutually authenticated DTLS as defined in the Functional Package for Transport Layer Security (TLS), version 2.1*
- *HTTPS*

] protocols to provide a communication channel between itself and another trusted IT product using certificates as defined in [*Functional Package for X.509, version 1.0*] that is logically distinct from other communication channels, provides assured identification of its end points, protects channel data from disclosure, and detects modification of the channel data.

Application Note: The intent of the mandatory portion of the above requirement is to use the cryptographic protocols identified in the requirement to establish and maintain a trusted channel between the TOE and an access point, VPN Gateway, or other trusted IT product.

The ST author must include the security functional requirements for the trusted channel protocol selected in FTP_ITC_EXT.1 in the main body of the ST.

Assured identification of endpoints is performed according to the authentication mechanisms used by the listed trusted channel protocols.

Claims from the Functional Package for X.509, version 1.0 are only required to the extent that they are needed to support the functionality required by the trusted protocols that are claimed.

The TSF is mandated to support mutually authenticated TLS, which means that it implements a protocol that requires the validation of a certificate presented by an external entity. Therefore, FIA_X509_EXT.1 and FIA_X509_EXT.2 will be claimed, as will FIA_TSM_EXT.1 for management of the trust store.

The requirement to implement mutually authenticated TLS also means that it

implements a protocol that requires the presentation of any certificates to an external entity. Therefore, FIA_XCU_EXT.2 will be claimed. FIA_X509_EXT.6 will also be claimed, along with any applicable dependencies, depending on how the certificates presented by the TOE are obtained.

FTP_ITC_EXT.1.2

The TSF shall permit the TSF to initiate communication via the trusted channel.

FTP_ITC_EXT.1.3

The TSF shall initiate communication via the trusted channel for wireless access point connections, administrative communication, configured enterprise connections, and [**selection**: *OTA updates*, *no other connections*].

Evaluation Activities

FTP_ITC_EXT.1
TSS
The evaluator shall examine the TSS to determine that it describes the details of the TOE connecting to access points, VPN Gateways, and other trusted IT products in terms of the cryptographic protocols specified in the requirement, along with TOE-specific options or procedures that might not be reflected in the specifications. The evaluator shall also confirm that all protocols listed in the TSS are specified and included in the requirements in the ST.

If OTA updates are selected, the TSS shall describe which trusted channel protocol is initiated by the TOE and is used for updates.

Guidance
The evaluator shall confirm that the operational guidance contains instructions for establishing the connection to access points, VPN Gateways, and other trusted IT products.

Tests
The evaluator shall also perform the following tests for each protocol listed:

- *Test FTP_ITC_EXT.1:1: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data are not sent in plaintext and that a protocol analyzer identifies the traffic as the protocol under testing.*
- *Test FTP_ITC_EXT.1:2: [conditional] If IPsec in accordance with the PP-Module for Virtual Private Network (VPN) Clients, version 3.0 is selected, the evaluator shall ensure that the TOE is able to initiate communications with a VPN Gateway, setting up the connections as described in the operational guidance and ensuring that communication is successful.*
- *Test FTP_ITC_EXT.1:3: [conditional]If OTA updates are selected, the evaluator shall trigger an update request according to the operational guidance and shall ensure that the communication is successful.*
- *Test FTP_ITC_EXT.1:4: For any other selected protocol (not tested in Test 1, 2, or 3), the evaluator shall ensure that the TOE is able to initiate communications with a trusted IT product using the protocol, setting up the connection as described in the operational guidance and ensuring that the communication is successful.*

5.1.10 Bluetooth

Bluetooth is a short-range wireless technology standard that is commonly used for exchanging data between devices over short distances. Most, if not all, mobile devices include Bluetooth hardware.

If this feature is implemented by the TOE, the following requirements must be claimed in the ST:

- **FDP_UPC_EXT.1/BLUETOOTH**

5.1.11

If this feature is implemented by the TOE, the following requirements must be claimed in the ST:

5.1.12

If this feature is implemented by the TOE, the following requirements must be claimed in the ST:

5.1.13 TOE Security Functional Requirements Rationale

The following rationale provides justification for each SFR for the TOE, showing that the SFRs are suitable to address the specified threats:

Table 21: SFR Rationale

Threat	Addressed by	Rationale
T.MALICIOUS_APP	FAU_GEN.1	FAU_GEN.1 helps mitigate the threat of a malicious app by logging evidence of potential malicious activity.
	FAU_SAR.1	FAU_SAR.1 helps mitigate the threat of a malicious app by providing a mechanism to read the audit trail.
	FAU_SEL.1 (Objective)	FAU_SEL.1 helps mitigate the threat of a network attack by allowing the TSF to determine the behavior that indicates potential malicious activity in the audit trail.
	FAU_STG.2	FAU_STG.2 helps mitigate the threat of a malicious app by maintaining the availability of the audit trail.
	FAU_STG.5	FAU_STG.5 helps mitigate the threat of a malicious app by maintaining the availability of the audit trail.
	FCS_RBG.6	FCS_RBG.6 helps mitigate the threat of a malicious app by providing a secure and legitimate interface that can be invoked by apps to perform DRBG functionality.
	FCS_SRV_EXT.1	FCS_SRV_EXT.1 helps mitigate the threat of a malicious app by providing a secure and legitimate interface that can be invoked by apps to perform cryptographic functionality.
	FCS_SRV_EXT.2 (Objective)	FCS_SRV_EXT.1 helps mitigate the threat of a malicious app by providing a secure and legitimate interface that can be invoked by apps to interact with protected storage.
	FDP_ACF_EXT.1	FDP_ACF_EXT.1 helps mitigate the threat of a malicious app by restricting the system services that are accessible to applications.
	FDP_ACF_EXT.2 (Selection-based)	FDP_ACF_EXT.2 helps mitigate the threat of a malicious app by providing separate copies of system resources for different application groups.
	FDP_ACF_EXT.3 (Objective)	FDP_ACF_EXT.3 helps mitigate the threat of a malicious app by enforcing policies on applications that prohibit write and execute permissions from being granted simultaneously.
	FDP_BCK_EXT.1 (Objective)	FDP_BCK_EXT.1 helps mitigate the threat of a malicious app by determining which data to include in backup operations.
	FDP_BLT_EXT.1 (Objective)	FDP_BLT_EXT.1 helps mitigate the threat of a malicious app by managing which applications communicate with Bluetooth devices.
	FMT_MOF_EXT.1	FMT_MOF_EXT.1 helps mitigate the threat of a malicious app by limiting the management functions that are available to a given user.
	FMT_SMF_EXT.1	FMT_SMF_EXT.1 helps mitigate the threat of a malicious app by defining the management functions that are provided by the TOE.
	FMT_SMF_EXT.2	FMT_SMF_EXT.2 helps mitigate the threat of a malicious app by ensuring that sensitive data is purged from the TOE when it is no longer enrolled in mobile device management.
	FMT_SMF_EXT.3 (Objective)	FMT_SMF_EXT.3 helps mitigate the threat of a malicious app by providing a current list of authorized administrators and their authorized management functions.
	FPT_AEX_EXT.1	FPT_AEX_EXT.1 helps mitigate the threat of a malicious app by using address space layout randomization for all applications.
	FPT_AEX_EXT.2	FPT_AEX_EXT.2 helps mitigate the threat of a malicious app by enforcing memory access permissions.
	FPT_AEX_EXT.3	FPT_AEX_EXT.3 helps mitigate the threat of a malicious app by implementing stack-based buffer overflow protection running on non-privileged domains.
	FPT_AEX_EXT.4	FPT_AEX_EXT.4 helps mitigate the threat of a malicious app by enforcing address space isolation.

	FPT_AEX_EXT.5 (Objective)	FPT_AEX_EXT.5 helps mitigate the threat of a malicious app by enforcing kernel level ASLR.
	FPT_AEX_EXT.6 (Objective)	FPT_AEX_EXT.6 helps mitigate the threat of a malicious app by preventing non-privileged execution domains from being writable and executable.
	FPT_AEX_EXT.7 (Objective)	FPT_AEX_EXT.7 helps mitigate the threat of a malicious app by enforcing heap-based buffer overflow protections.
	FPT_BBD_EXT.1 (Objective)	FPT_BBD_EXT.1 helps mitigate the threat of a malicious app by preventing the application processor's executable memory from being modified by the baseband processor.
	FPT_BLT_EXT.1 (Objective)	FPT_BLT_EXT.1 helps mitigate the threat of a malicious app by enforcing least functionality of the TOE's Bluetooth interface.
	FPT_NOT_EXT.1	FPT_NOT_EXT.1 helps mitigate the threat of a malicious app by failing to a secure state when self-test failures occur.
	FPT_NOT_EXT.2 (Objective)	FPT_NOT_EXT.2 helps mitigate the threat of a malicious app by enforcing remote attestation to ensure that the TSF has not been compromised.
	FPT_STM.1	FPT_STM.1 helps mitigate the threat of a malicious app by providing reliable time stamps for the audit trail.
	FPT_TST_EXT.1	FPT_TST_EXT.1 helps mitigate the threat of a malicious app by performing self-tests that may detect a compromise of the TSF.
	FPT_TST_EXT.2/POSTKERNEL (Objective)	FPT_TST_EXT.2/POSTKERNEL helps mitigate the threat of a malicious app by verifying the integrity of executable code after the kernel is loaded.
	FPT_TST_EXT.2/PREKERNEL	FPT_TST_EXT.2/PREKERNEL helps mitigate the threat of a malicious app by performing self-tests that may detect a compromise of the TSF.
	FPT_TST_EXT.3 (Selection-based)	FPT_TST_EXT.3 helps mitigate the threat of a malicious app by preventing the execution of untrusted application code.
	FPT_TUD_EXT.1	FPT_TUD_EXT.1 helps mitigate the threat of a malicious app by enforcing mechanisms that allow for the identification of the TSF and of applications running on it.
	FPT_TUD_EXT.2	FPT_TUD_EXT.2 helps mitigate the threat of a malicious app by enforcing mechanisms that allow for maintaining the authenticity and integrity of the TSF.
	FPT_TUD_EXT.3	FPT_TUD_EXT.3 helps mitigate the threat of a malicious app by enforcing mechanisms that allow for maintaining the authenticity and integrity of installed apps.
	FPT_TUD_EXT.4 (Selection-based)	FPT_TUD_EXT.4 helps mitigate the threat of a malicious app by preventing the execution of untrusted updates.
	FPT_TUD_EXT.5 (Objective)	FPT_TUD_EXT.5 helps mitigate the threat of a malicious app by enforcing installation of only trusted mobile application software.
	FPT_TUD_EXT.6 (Objective)	FPT_TUD_EXT.6 helps mitigate the threat of a malicious app by preventing intentional rollback of software updates.
T.NETWORK_ATTACK	FAU_GEN.1	FAU_GEN.1 helps mitigate the threat of a network attack by maintaining an audit trail of potential malicious activity.
	FAU_SAR.1	FAU_SAR.1 helps mitigate the threat of a network attack by providing a mechanism to read the audit trail.
	FAU_SEL.1 (Objective)	FAU_SEL.1 helps mitigate the threat of a network attack by allowing the TSF to determine the behavior that indicates potential malicious activity in the audit trail.
	FAU_STG.2	FAU_STG.2 helps mitigate the threat of a network attack by maintaining the availability of the audit trail.

FAU_STG.5	FAU_STG.5 helps mitigate the threat of a network attack by maintaining the availability of the audit trail.
FCS_CKM_EXT.6	FCS_CKM_EXT.6 helps mitigate the threat of network eavesdrop by using salts using where appropriate for cryptographic functions.
FCS_CKM_EXT.7/UNLOCKED (Implementation-based)	FCS_CKM_EXT.7/UNLOCKED helps mitigate the threat of a network attack by performing key establishment for trusted communications.
FCS_CKM.1/AKG	FCS_CKM.1/AKG helps mitigate the threat of a network attack by ensuring the generation of strong keys used for trusted communications.
FCS_CKM.1/SKG	FCS_CKM.1/SKG helps mitigate the threat of a network attack by ensuring the generation of strong keys used for trusted communications.
FCS_CKM.2	FCS_CKM.2 helps mitigate the threat of a network attack by implementing secure methods to perform key distribution for trusted communications.
FCS_COP.1/AEAD	FCS_COP.1/AEAD mitigates the threat of network attack by enforcing the use of secure authenticated encryption algorithms when wireless functionality is implemented by the TOE.
FCS_COP.1/Hash	FCS_COP.1/Hash helps mitigate the threat of a network attack by ensuring that secure hash algorithms are used for trusted communications.
FCS_COP.1/KeyedHash	FCS_COP.1/KeyedHash helps mitigate the threat of a network attack by ensuring that secure HMAC algorithms are used for trusted communications.
FCS_COP.1/SigGen	FCS_COP.1/SigGen helps mitigate the threat of a network attack by ensuring that secure digital signature algorithms are used for trusted communications.
FCS_COP.1/SigVer	FCS_COP.1/SigVer helps mitigate the threat of a network attack by ensuring that secure digital signature algorithms are used for trusted communications.
FCS_COP.1/SKC (Selection-based)	FCS_COP.1/SKC helps mitigate the threat of a network attack by ensuring that secure symmetric algorithms are used for trusted communications.
FCS_HTTPS_EXT.1	FCS_HTTPS_EXT.1 helps mitigate the threat of a network attack by implementing a secure protocol (HTTPS) for trusted communications.
FCS_RBG_EXT.2 (Objective)	FCS_RBG_EXT.2 helps mitigate the threat of a network attack by ensuring the DRBG state at power-off and startup remains the same.
FCS_RBG.1	FCS_RBG.1 helps mitigate the threat of a network attack by ensuring that keys used for trusted communications are generated using a secure DRBG.
FCS_RBG.2 (Selection-based)	FCS_RBG.2 helps mitigate the threat of network eavesdrop by ensuring that the TOE's DRBG is seeded with sufficient entropy to ensure the generation of strong cryptographic keys.
FCS_RBG.3 (Selection-based)	FCS_RBG.3 helps mitigate the threat of a network attack by ensuring that the TOE's DRBG is seeded with sufficient entropy to ensure the generation of strong cryptographic keys.
FCS_RBG.4 (Selection-based)	FCS_RBG.4 helps mitigate the threat of a network attack by ensuring that the TOE's DRBG is seeded with sufficient entropy to ensure the generation of strong cryptographic keys.
FCS_RBG.5 (Selection-based)	FCS_RBG.5 helps mitigate the threat of a network attack by ensuring that the TOE's DRBG is seeded with sufficient entropy to ensure the generation of strong cryptographic keys.
FCS_RBG.6	FCS_RBG.6 helps mitigate the threat of a network attack by providing a secure DRBG service for third-party applications running on the TOE which may use this service to generate their own cryptographic keys

		for trusted communications.
	FCS_SRV_EXT.1	FCS_SRV_EXT.1 helps mitigate the threat of a network attack by ensuring applications have the proper mechanism to perform cryptographic operations.
	FCS_SRV_EXT.2 (Objective)	FCS_SRV_EXT.2 helps mitigate the threat of a network attack by ensuring applications have the proper mechanism to perform cryptographic operations with keys in secure storage.
	FDP_BLT_EXT.1 (Objective)	FDP_BLT_EXT.1 helps mitigate the threat of a network attack by managing which applications communicate with Bluetooth devices.
	FDP_IFC_EXT.1	FDP_IFC_EXT.1 helps mitigate the threat of a network attack by ensuring that the TOE has the ability to enforce the use of an IPsec VPN for all network traffic.
	FDP_STG_EXT.1	FDP_STG_EXT.1 helps mitigate the threat of network eavesdrop by protecting the X.509 certificates used for trusted communications.
	FDP_UPC_EXT.1/APPS	FDP_UPC_EXT.1/APPS helps mitigate the threat of network eavesdrop by implementing a means for applications to communicate securely via a trusted channel.
	FDP_UPC_EXT.1/BLUETOOTH (Implementation-based)	FDP_UPC_EXT.1/BLUETOOTH helps mitigate the threat of a network attack by protecting communication channels using Bluetooth functionality via a trusted channel.
	FMT_MOF_EXT.1	FMT_MOF_EXT.1 helps mitigate the threat of a network attack by limiting the management functions that are available to a given user.
	FMT_SMF_EXT.1	FMT_SMF_EXT.1 helps mitigate the threat of a network attack by providing a list of specific management functions.
	FMT_SMF_EXT.2	FMT_SMF_EXT.2 helps mitigate the threat of a network attack by specifying the remediation actions allowed upon enrollment.
	FPT_BLT_EXT.1 (Objective)	FPT_BLT_EXT.1 helps mitigate the threat of a network attack by enforcing least functionality of the TOE's Bluetooth interface.
	FPT_FLS.1	FPT_FLS.1 helps mitigate the threat of a network attack by ensuring that a malfunctioning DRBG function cannot be used to generate potentially insecure keys.
	FPT_STM.1	FPT_STM.1 helps mitigate the threat of a malicious app by providing reliable time stamps for the audit trail.
	FPT_TST.1	FPT_TST.1 helps mitigate the threat of a network attack by implementing a mechanism to detect when the DRBG may be failing to generate secure cryptographic keys.
	FTA_SSL_EXT.1	FTA_SSL_EXT.1 helps mitigate the threat of a network attack by managing the transition to a locked state after a set time or operation.
	FTA_TAB.1	FTA_TAB.1 helps mitigate the threat of a network attack by providing actionable consequences for misuse of the TSF.
	FTP_ITC_EXT.1	FTP_ITC_EXT.1 helps mitigate the threat of a network attack by requiring the TSF to implement trusted protocols for network communication.
T.NETWORK_EAVESDROP	FCS_CKM_EXT.6	FCS_CKM_EXT.6 helps mitigate the threat of network eavesdrop by using salts using where appropriate for cryptographic functions.
	FCS_CKM_EXT.7/UNLOCKED (Implementation-based)	FCS_CKM_EXT.7/UNLOCKED helps mitigate the threat of network eavesdrop by performing key establishment for trusted communications.
	FCS_CKM.1/AKG	FCS_CKM.1/AKG helps mitigate the threat of network eavesdrop by ensuring the generation of strong keys used for trusted communications.

FCS_CKM.1/SKG	FCS_CKM.1/SKG helps mitigate the threat of network eavesdrop by ensuring the generation of strong keys used for trusted communications.
FCS_CKM.2	FCS_CKM.2 helps mitigate the threat of a network eavesdrop by implementing secure methods to perform key distribution for trusted communications.
FCS_COP.1/AEAD	FCS_COP.1/AEAD mitigates the threat of network eavesdrop by enforcing the use of a cryptographic algorithm to protect data in transit that includes assurance of both authenticity and confidentiality.
FCS_COP.1/Hash	FCS_COP.1/Hash helps mitigate the threat of network eavesdrop by ensuring that secure hash algorithms are used for trusted communications.
FCS_COP.1/KeyedHash	FCS_COP.1/KeyedHash helps mitigate the threat of a network eavesdrop by ensuring that secure HMAC algorithms are used for trusted communications.
FCS_COP.1/SigGen	FCS_COP.1/SigGen helps mitigate the threat of network eavesdrop by ensuring that secure digital signature algorithms are used for trusted communications.
FCS_COP.1/SigVer	FCS_COP.1/SigVer helps mitigate the threat of network eavesdrop by ensuring that secure digital signature algorithms are used for trusted communications.
FCS_COP.1/SKC (Selection-based)	FCS_COP.1/SKC helps mitigate the threat of network eavesdrop by ensuring that secure symmetric algorithms are used for trusted communications.
FCS_HTTPS_EXT.1	FCS_HTTPS_EXT.1 helps mitigate the threat of network eavesdrop by implementing a secure protocol (HTTPS) for trusted communications.
FCS_RBG_EXT.2 (Objective)	FCS_RBG_EXT.2 helps mitigate the threat of network eavesdrop by defining requirements for secure DRBG implementation.
FCS_RBG_EXT.3 (Objective)	FCS_RBG_EXT.3 helps mitigate the threat of network eavesdrop by defining requirements for secure DRBG implementation.
FCS_RBG.2 (Selection-based)	FCS_RBG.2 helps mitigate the threat of network eavesdrop by ensuring that the TOE's DRBG is seeded with sufficient entropy to ensure the generation of strong cryptographic keys.
FCS_RBG.3 (Selection-based)	FCS_RBG.3 helps mitigate the threat of network eavesdrop by ensuring that the TOE's DRBG is seeded with sufficient entropy to ensure the generation of strong cryptographic keys.
FCS_RBG.4 (Selection-based)	FCS_RBG.4 helps mitigate the threat of network eavesdrop by ensuring that the TOE's DRBG is seeded with sufficient entropy to ensure the generation of strong cryptographic keys.
FCS_RBG.5 (Selection-based)	FCS_RBG.5 helps mitigate the threat of network eavesdrop by ensuring that the TOE's DRBG is seeded with sufficient entropy to ensure the generation of strong cryptographic keys.
FCS_RBG.6	FCS_RBG.6 helps mitigate the threat of a network eavesdrop by providing a secure DRBG service for third-party applications running on the TOE which may use this service to generate their own cryptographic keys for trusted communications.
FCS_SRV_EXT.1	FCS_SRV_EXT.1 helps mitigate the threat of network eavesdrop by ensuring applications have the proper mechanism to perform cryptographic operations.
FDP_BLT_EXT.1 (Objective)	FDP_BLT_EXT.1 helps mitigate the threat of network eavesdrop by managing which applications communicate with Bluetooth devices.
FDP_IFC_EXT.1	FDP_IFC_EXT.1 helps mitigate the threat of network eavesdrop by ensuring that the TOE has the ability to enforce the use of an IPsec VPN for all network traffic.
FDP_STG_EXT.1	FDP_STG_EXT.1 helps mitigate the threat of network

		eavesdrop by protecting the X.509 certificates used for trusted communications.
	FDP_UPC_EXT.1/APPS	FDP_UPC_EXT.1/APPS helps mitigate the threat of network eavesdrop by implementing a means for applications to communicate securely via a trusted channel.
	FDP_UPC_EXT.1/BLUETOOTH (Implementation-based)	FDP_UPC_EXT.1/BLUETOOTH helps mitigate the threat of network eavesdrop by protecting communication channels using Bluetooth functionality via a trusted channel.
	FIA_X509_EXT.6	FIA_X509_EXT.6 helps mitigate the threat of network eavesdrop by defining a secure mechanism by which the TSF can acquire certificates for its own use.
	FPT_BLT_EXT.1 (Objective)	FPT_BLT_EXT.1 helps mitigate the threat of network eavesdrop by enforcing least functionality of the TOE's Bluetooth interface.
	FPT_FLS.1	FPT_FLS.1 helps mitigate the threat of network eavesdrop by ensuring that a malfunctioning DRBG function cannot be used to generate potentially insecure keys.
	FPT_TST.1	FPT_TST.1 helps mitigate the threat of network eavesdrop by implementing a mechanism to detect when the DRBG may be failing to generate secure cryptographic keys.
	FTP_ITC_EXT.1	FTP_ITC_EXT.1 helps mitigate the threat of network eavesdrop by requiring the TSF to implement trusted protocols for network communication.
T.PERSISTENT_PRESENCE	FMT_MOF_EXT.1	FMT_MOF_EXT.1 helps mitigate the threat of persistent presence by enforcing limitations on TSF usage when it is enrolled in mobile device management.
	FMT_SMF_EXT.1	FMT_SMF_EXT.1 helps mitigate the threat of persistent presence by defining the management functions that are supported by the TOE.
	FMT_SMF_EXT.2	FMT_SMF_EXT.2 helps mitigate the threat of persistent presence by maintaining a secure state of the TOE if it is unenrolled from mobile device management.
	FPT_NOT_EXT.1	FPT_NOT_EXT.1 helps mitigate the threat of persistent presence by having the TOE enter a secure failure state when self-test integrity failures occur.
	FPT_TST_EXT.1	FPT_TST_EXT.1 helps mitigate the threat of a persistent presence by performing self-tests to verify the integrity of the TSF.
	FPT_TST_EXT.2/PREKERNEL	FPT_TST_EXT.2/PREKERNEL helps mitigate the threat of a persistent presence by performing self-tests to verify the integrity of the TSF.
	FPT_TST_EXT.3 (Selection-based)	FPT_TST_EXT.3 helps mitigate the threat of a persistent presence by preventing the execution of untrusted application code.
	FPT_TUD_EXT.2	FPT_TUD_EXT.2 helps mitigate the threat of a persistent presence by ensuring that only legitimate updates are applied to the TOE.
	FPT_TUD_EXT.3	FPT_TUD_EXT.3 helps mitigate the threat of a persistent presence by ensuring mobile application software updates are digitally signed prior to installation.
	FPT_TUD_EXT.4 (Selection-based)	FPT_TUD_EXT.4 helps mitigate the threat of a persistent presence by preventing the execution of untrusted updates.
	FPT_TUD_EXT.5 (Objective)	FPT_TUD_EXT.5 helps mitigate the threat of a persistent presence by enforcing mobile application software verification by X.509v3 prior to installation.
	FPT_TUD_EXT.6 (Objective)	FPT_TUD_EXT.6 helps mitigate the threat of a persistent presence by preventing intentional rollback of software updates.

T.PHYSICAL_ACCESS	FCS_CKM_EXT.1	FCS_CKM_EXT.1 helps mitigate the threat of physical access by implementing a method of protecting data at rest while the TOE is in a locked state.
	FCS_CKM_EXT.3	FCS_CKM_EXT.3 helps mitigate the threat of physical access by implementing a method of protecting data at rest while the TOE is in a locked state.
	FCS_CKM_EXT.5	FCS_CKM_EXT.5 helps mitigate the threat of physical access by implementing a secure mechanism to erase protected data from the TOE.
	FCS_CKM_EXT.6	FCS_CKM_EXT.6 helps mitigate the threat of physical access by generating salts using RBG.
	FCS_CKM_EXT.7/LOCKED	FCS_CKM_EXT.7/LOCKED helps mitigate the threat of physical access by implementing a method of protecting data at rest while the TOE is in a locked state.
	FCS_CKM_EXT.7/UNLOCKED (Implementation-based)	FCS_CKM_EXT.7/UNLOCKED helps mitigate the threat of physical access by implementing a method of protecting data at rest while the TOE is in a locked state.
	FCS_CKM_EXT.8	FCS_CKM_EXT.8 helps mitigate the threat of physical access by performing password conditioning to increase the difficulty of deriving a key from a password.
	FCS_CKM.2	FCS_CKM.2 helps mitigate the threat of physical access by implementing secure methods to perform key distribution for trusted communications.
	FCS_CKM.6	FCS_CKM.6 helps mitigate the threat of physical access by ensuring that keys used for trusted communications are destroyed in a secure manner.
	FCS_COP.1/Hash	FCS_COP.1/Hash helps mitigate the threat of physical access by implementing an integrity mechanism used to verify stored keys.
	FCS_COP.1/KeyedHash	FCS_COP.1/KeyedHash helps mitigate the threat of physical access by implementing an integrity mechanism used to verify stored keys.
	FCS_IV_EXT.1	FCS_IV_EXT.1 helps mitigate the threat of physical access by ensuring that appropriate IVs are used for symmetric keys.
	FCS_RBG.1	FCS_RBG.1 helps mitigate the threat of physical access by ensuring that keys used for protected storage are generated using a secure DRBG.
	FCS_RBG.2 (Selection-based)	FCS_RBG.2 helps mitigate the threat of network eavesdrop by ensuring that the TOE's DRBG is seeded with sufficient entropy to ensure the generation of strong cryptographic keys.
	FCS_RBG.3 (Selection-based)	FCS_RBG.3 helps mitigate the threat of physical access by ensuring that the TOE's DRBG is seeded with sufficient entropy to ensure the generation of strong cryptographic keys.
	FCS_RBG.4 (Selection-based)	FCS_RBG.4 helps mitigate the threat of physical access by ensuring that the TOE's DRBG is seeded with sufficient entropy to ensure the generation of strong cryptographic keys.
	FCS_RBG.5 (Selection-based)	FCS_RBG.5 helps mitigate the threat of physical access by ensuring that the TOE's DRBG is seeded with sufficient entropy to ensure the generation of strong cryptographic keys.
	FCS_STG_EXT.1	FCS_STG_EXT.1 helps mitigate the threat of physical access by implementing a secure key storage for keys used to protect data at rest.
	FCS_STG_EXT.2	FCS_STG_EXT.2 helps mitigate the threat of physical access by enforcing confidentiality for key storage.
	FCS_STG_EXT.3	FCS_STG_EXT.3 helps mitigate the threat of physical access by enforcing integrity for key storage.
	FDP_DAR_EXT.1	FDP_DAR_EXT.1 helps mitigate the threat of physical access by encrypting sensitive data at rest.

FDP_DAR_EXT.2	FDP_DAR_EXT.2 helps mitigate the threat of physical access by allowing data to be marked as sensitive such that it is encrypted while at rest.
FIA_AFL_EXT.1	FIA_AFL_EXT.1 helps mitigate the threat of physical access by limiting the extent to which brute force authentication attempts to the TOE can be made.
FIA_PMG_EXT.1	FIA_PMG_EXT.1 helps mitigate the threat of physical access by defining strong password characteristics.
FIA_TRT_EXT.1	FIA_AFL_EXT.1 helps mitigate the threat of physical access by limiting the extent to which brute force authentication attempts to the TOE can be made.
FIA_UAU_EXT.1	FIA_UAU_EXT.1 helps mitigate the threat of physical access by preventing decryption prior to proper authorization to the device.
FIA_UAU_EXT.2	FIA_UAU_EXT.2 helps mitigate the threat of physical access by requiring successful authentication before allowing the user to take action on the TOE.
FIA_UAU_EXT.4 (Optional)	FIA_UAU_EXT.4 helps mitigate the threat of physical access by enforcing a secondary authentication mechanism for accessing enterprise resources.
FIA_UAU.5	FIA_UAU.5 helps mitigate the threat of physical access by defining the supported authentication mechanisms.
FIA_UAU.6/CREDENTIAL	FIA_UAU.6/CREDENTIAL helps mitigate the threat of physical access by implementing functionality to require re-authentication before any attempted changes to authenticated mechanisms.
FIA_UAU.6/LOCKED	FIA_UAU.6/LOCKED helps mitigate the threat of physical access by implementing functionality to require re-authentication in certain conditions.
FIA_UAU.7	FIA_UAU.7 helps mitigate the threat of physical access by providing only limited information during authentication.
FPT_FLS.1	FPT_FLS.1 helps mitigate the threat of physical access by ensuring that a malfunctioning DRBG function cannot be used to generate potentially insecure keys.
FPT_JTA_EXT.1	FPT_JTA_EXT.1 helps mitigate the threat of physical access by specifying the mechanism used to control access to JTAG.
FPT_KST_EXT.1	FPT_KST_EXT.1 helps mitigate the threat of physical access by ensuring plaintext key material is not stored in readable non-volatile memory.
FPT_KST_EXT.2	FPT_KST_EXT.2 helps mitigate the threat of physical access by preventing transmission of plaintext key material outside the secure boundary of the TOE.
FPT_KST_EXT.3	FPT_KST_EXT.3 helps mitigate the threat of physical access by ensuring TOE users cannot export plaintext keys.
FPT_TST.1	FPT_TST.1 helps mitigate the threat of physical access by implementing a mechanism to detect when the DRBG may be failing to generate secure cryptographic keys.
FTA_SSL_EXT.1	FTA_SSL_EXT.1 helps mitigate the threat of physical access by managing the transition to a locked state after a set time or operation.

5.2 Security Assurance Requirements

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

This section lists the set of SARs from CC part 3 that are required in evaluations against this PP. Individual Evaluation Activities to be performed are specified both in as well as in this section.

The general model for evaluation of TOEs against STs written to conform to this PP is as follows:

After the ST has been approved for evaluation, the ITSEF will obtain the TOE, supporting environmental IT, and the administrative/user guides for the TOE. The ITSEF is expected to perform actions mandated by the Common Evaluation Methodology (CEM) for the ASE and ALC SARs. The ITSEF also performs the Evaluation Activities contained within , which are intended to be an interpretation of the other CEM evaluation requirements as they apply to the specific technology instantiated in the TOE. The Evaluation Activities that are captured in also provide clarification as to what the developer needs to provide to demonstrate the TOE is compliant with the PP.

The TOE Security Assurance Requirements are identified in Table 22.

Table 22: Security Assurance Requirements

Assurance Class	Assurance Components
Security Target (ASE)	Conformance Claims (ASE_CCL.1)
	Extended Components Definition (ASE_ECD.1)
	ST Introduction (ASE_INT.1)
	Security Objectives for the Operational Environment (ASE_OBJ.1)
	Stated Security Requirements (ASE_REQ.1)
	Security Problem Definition (ASE_SPD.1)
	TOE Summary Specification (ASE_TSS.1)
Development (ADV)	Basic Functional Specification (ADV_FSP.1)
Guidance Documents (AGD)	Operational User Guidance (AGD_OPE.1)
	Preparative Procedures (AGD_PRE.1)
Life Cycle Support (ALC)	Labeling of the TOE (ALC_CMC.1)
	TOE CM Coverage (ALC_CMS.1)
	Timely Security Updates (ALC_TSU_EXT)
Tests (ATE)	Independent Testing – Sample (ATE_IND.1)
Vulnerability Assessment (AVA)	Vulnerability Survey (AVA_VAN.1)

5.2.1 Class ASE: Security Target

The ST is evaluated as per ASE activities defined in the CEM for ASE_CCL.1, ASE_ECD.1, ASE_INT.1, ASE_OBJ.1, ASE_REQ.1, ASE_SPD.1, and ASE_TSS.1. In addition, there may be Evaluation Activities specified within that call for necessary descriptions to be included in the TSS that are specific to the TOE technology type.

5.2.2 Class ADV: Development

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

ADV_FSP.1 Basic Functional Specification

Developer action elements:

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.

Application Note: As indicated in the introduction to this section, the functional specification is comprised of the information contained in the AGD_OPE, AGD_PRE, and the API information that is provided to application developers, including the APIs that require privilege to invoke.

The developer may reference a website accessible to application developers and the evaluator. The API documentation must include those interfaces required in this profile. The API documentation must clearly indicate to which products and versions each available function applies.

The evaluation activities in the functional requirements point to evidence that should exist in the documentation and TSS section; since these are directly associated with the SFRs, the tracing in element ADV_FSP.1.2D is implicitly already done and no additional documentation is necessary.

Content and presentation elements:

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.

Evaluator action elements:

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.

Evaluation Activities

ADV_FSP.1
There are no specific evaluation activities associated with these SARs, except ensuring the information is provided. The functional specification documentation is provided to support the evaluation activities described in , and other activities described for AGD, ATE, and AVA SARs. The requirements on the content of the functional specification information is implicitly assessed by virtue of the other evaluation activities being performed; if the evaluator is unable to perform an activity because there is insufficient interface information, then an adequate functional specification has not been provided.

5.2.3 Class AGD: Guidance Documentation

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

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

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

Guidance pertaining to particular security functionality is also provided; requirements on such guidance are contained in the evaluation activities specified with each requirement.

AGD_OPE.1 Operational User Guidance

Developer action elements:

AGD_OPE.1.1D

The developer shall provide operational user guidance.

Application Note: The operational user guidance does not have to be contained in a single document. Guidance to users, administrators and application developers can be spread among documents or web pages. Where appropriate, the guidance documentation is expressed in the eXtensible Configuration Checklist Description Format (XCCDF) to support security automation.

Rather than repeat information here, the developer should review the evaluation activities for this component to ascertain the specifics of the guidance that the evaluator will be checking for. This will provide the necessary information for the preparation of acceptable guidance.

Content and presentation elements:

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.

Application Note: User and administrator (e.g., MDM agent), and application developer are to be considered in the definition of user role.

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 OS (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.

Evaluator action elements:

AGD_OPE.1.1E

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

Evaluation Activities

AGD_OPE.1
Some of the contents of the operational guidance are verified by the evaluation activities in and evaluation of the TOE according to the . The following additional information is also required.

The operational guidance shall contain a list of natively installed applications and any relevant version numbers. If any third party vendors are permitted to install applications before purchase by the end user or enterprise, these applications shall also be listed.

The operational guidance shall contain instructions for configuring the cryptographic engine associated with the evaluated configuration of the TOE. It shall provide a warning to the administrator that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.

The documentation must describe the process for verifying updates to the TOE by verifying a digital signature. The evaluator shall verify that this process includes the following steps:

- *Instructions for obtaining the update itself. This should include instructions for making the update accessible to the TOE (e.g., placement in a specific directory).*
- *Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes generation of the hash/digital signature.*

The TOE will likely contain security functionality that does not fall in the scope of evaluation under this PP. The operational guidance shall make it clear to an administrator which security functionality is covered by the evaluation activities.

AGD_PRE.1 Preparative Procedures

Developer action elements:

AGD_PRE.1.1D

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

Application Note: As with the operational guidance, the developer should look to the evaluation activities to determine the required content with respect to preparative procedures.

Content and presentation elements:

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.

Evaluator action elements:

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 OS can be prepared securely for operation.

Evaluation Activities

AGD_PRE.1
As indicated in the introduction above, there are significant expectations with respect to the documentation—especially when configuring the operational environment to support TOE functional requirements. The evaluator shall check to ensure that the guidance provided for the TOE adequately addresses all platforms claimed for the TOE in the ST.

5.2.4 Class ALC: Life-cycle Support

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

ALC_CMC.1 Labeling of the TOE

Developer action elements:

ALC_CMC.1.1D

The developer shall provide the TOE and a reference for the TOE.

Content and presentation elements:

ALC_CMC.1.1C

The TOE shall be labeled with a unique reference.

Evaluator action elements:

ALC_CMC.1.1E

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

Evaluation Activities

ALC_CMC.1
The evaluator shall check the ST to ensure that it contains an identifier (such as a product name/version number) that specifically identifies the version that meets the requirements of the ST. Further, the evaluator shall check the AGD guidance and TOE samples received for testing to ensure that the version number is consistent with that in the ST. If the vendor maintains a web site advertising the TOE, the evaluator shall examine the information on the web site to ensure that the information in the ST is sufficient to distinguish the product.

ALC_CMS.1 TOE CM Coverage

Developer action elements:

ALC_CMS.1.1D

The developer shall provide a configuration list for the TOE.

Content and presentation elements:

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.

Evaluator action elements:

ALC_CMS.1.1E

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

Application Note: The "evaluation evidence required by the SARs" in this PP is limited to the information in the ST coupled with the guidance provided to administrators and users under the AGD requirements. By ensuring that the TOE is specifically identified and that this identification is consistent in the ST and in the AGD guidance (as done in the evaluation activity for ALC_CMC.1), the

evaluator implicitly confirms the information required by this component.

Life-cycle support is targeted aspects of the developer's life-cycle and instructions to providers of applications for the developer's devices, rather than an in-depth examination of the TSF manufacturer's development and configuration management process. This is not meant to diminish the critical role that a developer's practices play in contributing to the overall trustworthiness of a product; rather, it's a reflection on the information to be made available for evaluation.

Evaluation Activities

ALC_CMS.1
The evaluator shall ensure that the developer has identified (in public-facing development guidance for their platform) one or more development environments appropriate for use in developing applications for the developer's platform. For each of these development environments, the developer shall provide information on how to configure the environment to ensure that buffer overflow protection mechanisms in the environments are invoked (e.g., compiler and linker flags). The evaluator shall ensure that this documentation also includes an indication of whether such protections are on by default, or have to be specifically enabled.

The evaluator shall ensure that the TSF is uniquely identified (with respect to other products from the TSF vendor), and that documentation provided by the developer in association with the requirements in the ST is associated with the TSF using this unique identification.

ALC_TSU_EXT.1 Timely Security Updates

Developer action elements:

ALC_TSU_EXT.1.1D

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

Content and presentation elements:

ALC_TSU_EXT.1.1C

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

Application Note: The software to be described includes the operating systems of the application processor and the baseband processor, as well as any firmware and applications. The process description includes the TOE developer processes as well as any third-party (carrier) processes. The process description includes each deployment mechanism (e.g., over-the-air updates, per-carrier updates, downloaded updates).

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.

Application Note: The total length of time may be presented as a summation of the periods of time that each party (e.g., TOE developer, mobile carrier) on the critical path consumes. The time period until public availability per deployment mechanism may differ; each is described.

ALC_TSU_EXT.1.3C

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

Application Note: The reporting mechanism could include web sites, email addresses, as well as a means to protect the sensitive nature of the report (e.g., public keys that could be used to encrypt the details of a proof-of-concept exploit).

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.

Application Note: The purpose of providing this information is so that users and enterprises can determine which devices are susceptible to publicly known vulnerabilities so that they can make appropriate risk decisions, such as limiting access to enterprise resources until updates are installed.

Evaluator action elements:

ALC_TSU_EXT.1.1E

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

Evaluation Activities

ALC_TSU_EXT.1
The evaluator shall verify that the TSS contains a description of the timely security update process used by the developer to create and deploy security updates. The evaluator shall verify that this description addresses the TOE OS, the firmware, and bundled applications, each. The evaluator shall also verify that, in addition to the TOE developer's process, any carrier or other third-party processes are also addressed in the description. The evaluator shall also verify that each mechanism for deployment of security updates is described.

The evaluator shall verify that, for each deployment mechanism described for the update process, the TSS lists a time between public disclosure of a vulnerability and public availability of the security update to the TOE patching this vulnerability, to include any third-party or carrier delays in deployment. The evaluator shall verify that this time is expressed in a number or range of days.

The evaluator shall verify that this description includes the publicly available mechanisms (including either an email address or website) for reporting security issues related to the TOE. The evaluator shall verify that the description of this mechanism includes a method for protecting the report either using a public key for encrypting email or a trusted channel for a website.

The evaluator shall verify that the description includes where users can seek information about the availability of new security updates including details of the specific public vulnerabilities corrected by each update. The evaluator shall verify that the description includes the minimum amount of time that the TOE is expected to be supported with security updates, and the process by which users can seek information about when the TOE is no longer expected to receive security updates.

5.2.5 Class ATE: Tests

Testing is specified for functional aspects of the system as well as aspects that take advantage of design or implementation weaknesses. The former is done through the ATE_IND family, while the latter is through the AVA_VAN family. At the assurance level specified in this PP, testing is based on advertised functionality and interfaces with dependency on the availability of design information. One of the primary outputs of the evaluation process is the test report as specified in the following requirements.

Since many of the APIs are not exposed at the user interface (e.g., touch screen), the ability to stimulate the necessary interfaces requires a developer's test environment. This test environment will allow the evaluator, for example, to access APIs and view file system information that is not available on consumer Mobile Devices.

ATE_IND.1 Independent Testing – Conformance

Developer action elements:

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

Content and presentation elements:

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

Evaluator action elements:

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.

Evaluation Activities

ATE_IND.1
The evaluator shall prepare a test plan and report documenting the testing aspects of the system. The test plan covers all of the testing actions contained in the and the body of this PP's Evaluation Activities. While it is not necessary to have one test case per test listed in an evaluation activity, the evaluator must document in the test plan that each applicable testing requirement in the ST is covered.

The test plan identifies the platforms to be tested, and for those platforms not included in the test plan but included in the ST, the test plan provides a justification for not testing the platforms. This justification must address the differences between the tested platforms and the untested platforms, and make an argument that the differences do not affect the testing to be performed. It is not sufficient to merely assert that the differences have no affect; rationale must be provided. If all platforms claimed in the ST are tested, then no rationale is necessary.

The test plan describes the composition of each platform to be tested, and any setup that is necessary beyond what is contained in the AGD documentation. It should be noted that the evaluator is expected to follow the AGD documentation for installation and setup of each platform either as part of a test or as a standard pre-test condition. This may include special test drivers or tools. For each driver or tool, an argument (not just an assertion) should be provided that the driver or tool will not adversely affect the performance of the functionality by the TOE and its platform. This also includes the configuration of the cryptographic engine to be used. The cryptographic algorithms implemented by this engine are those specified by this PP and used by the cryptographic protocols being evaluated (IPsec, TLS/HTTPS, SSH).

The test plan identifies high-level test objectives as well as the test procedures to be followed to achieve those objectives. These procedures include expected results. The test report (which could just be an annotated version of the test plan) details the activities that took place when the test procedures were executed, and includes the actual results of the tests. This shall be a cumulative account, so if there was a test run that resulted in a failure; a fix installed; and then a successful re-run of the test, the report would show a "fail" and "pass" result (and the supporting details), and not just the "pass" result.

5.2.6 Class AVA: Vulnerability Assessment

For the current generation of this protection profile, the evaluation lab is expected to survey open sources to discover what vulnerabilities have been discovered in these types of products. In most cases, these vulnerabilities will require sophistication beyond that of a basic attacker. Until penetration tools are created and uniformly distributed to the evaluation labs, the evaluator will not be expected to test for these vulnerabilities in the TOE. The labs will be expected to comment on the likelihood of these vulnerabilities given the documentation provided by the vendor. This information will be used in the development of penetration testing tools and for the development of future protection profiles.

AVA_VAN.1 Vulnerability Survey

Developer action elements:

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

Content and presentation elements:

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

Evaluator action elements:

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.

Application Note: Public domain sources include the Common Vulnerabilities and Exposures (CVE) dictionary for publicly-known vulnerabilities.

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.

Evaluation Activities

AVA_VAN.1
The evaluator shall generate a report to document their findings with respect to this requirement. This report could physically be part of the overall test report mentioned in ATE_IND, or a separate document. The evaluator performs a search of public information to find vulnerabilities that have been found in mobile devices and the implemented communication protocols in general, as well as those that pertain to the particular TOE. The evaluator documents the sources consulted and the vulnerabilities found in the report.

For each vulnerability found, the evaluator either provides a rationale with respect to its non-applicability, or the evaluator formulates a test (using the guidelines provided in ATE_IND) to confirm the vulnerability, if suitable. Suitability is determined by assessing the attack vector needed to take advantage of the vulnerability. If exploiting the vulnerability requires expert skills and an electron microscope, for instance, then a test would not be suitable and an appropriate justification would be formulated.

Appendix A - Implementation-dependent Requirements

Implementation-dependent Requirements Appendix defines requirements that must be claimed in the ST if the TOE implements particular product features. For this technology type, the following product features require the claiming of additional SFRs:

A.0.1 Bluetooth

Bluetooth is a short-range wireless technology standard that is commonly used for exchanging data between devices over short distances. Most, if not all, mobile devices include Bluetooth hardware.

If this feature is implemented by the TOE, the following requirements must be claimed in the ST:

- **FDP_UPC_EXT.1/BLUETOOTH**

A.0.2

If this feature is implemented by the TOE, the following requirements must be claimed in the ST:

A.0.3

If this feature is implemented by the TOE, the following requirements must be claimed in the ST:

Appendix B - Extended Component Definitions

This appendix contains the definitions for all extended requirements specified in the PP.

B.1 Extended Components Table

All extended components specified in the PP are listed in this table:

Table 23: Extended Component Definitions

Functional Class	Functional Components
Class: Cryptographic Support (FCS)	FCS_CKM_EXT Cryptographic Key Management FCS_HTTPS_EXT HTTPS Protocol FCS_IV_EXT Initialization Vector Generation FCS_RBG_EXT Random Bit Generation FCS_SRV_EXT Cryptographic Algorithm Services FCS_STG_EXT Cryptographic Key Storage
Class: Identification and Authentication (FIA)	FIA_AFL_EXT Authentication Failures FIA_PMG_EXT Password Management FIA_TRT_EXT Authentication Throttling FIA_UAU_EXT User Authentication FIA_X509_EXT X.509 Certificates
Class: Protection of the TSF (FPT)	FPT_AEX_EXT Anti-Exploitation Capabilities FPT_BBD_EXT Baseband Processing FPT_BLT_EXT Limitation of Bluetooth Profile Support FPT_JTA_EXT JTAG Disablement FPT_KST_EXT Key Storage FPT_NOT_EXT Self-Test Notification FPT_TST_EXT TSF Self Test FPT_TUD_EXT TSF Updates
Class: Security Management (FMT)	FMT_MOF_EXT Management of Functions in TSF FMT_SMF_EXT Specification of Management Functions
Class: TOE Access (FTA)	FTA_SSL_EXT Session Locking and Termination
Class: Trusted Path/Channels (FTP)	FTP_ITC_EXT Inter-TSF Trusted Channel
Class: User Data Protection (FDP)	FDP_ACF_EXT Access Control Functions FDP_BCK_EXT Application Backup FDP_BLT_EXT Limitation of Bluetooth Device Access FDP_DAR_EXT Data-at-Rest Encryption FDP_IFC_EXT Subset Information Flow Control FDP_STG_EXT User Data Storage FDP_UPC_EXT Inter-TSF User Data Transfer Protection

B.2 Extended Component Definitions

B.2.1 Class: Cryptographic Support (FCS)

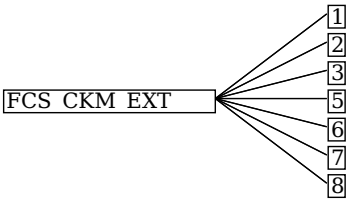
This PP defines the following extended components as part of the FCS class originally defined by CC Part 2:

B.2.1.1 FCS_CKM_EXT Cryptographic Key Management

Family Behavior

This family defines requirements for management of cryptographic keys that are not addressed by FCS_CKM in CC Part 2.

Component Leveling



FCS_CKM_EXT.1, Cryptographic Key Support, requires the TSF to implement a Root Encryption Key (REK).

FCS_CKM_EXT.2, Cryptographic Key Random Generation, requires the TSF to specify the mechanism it uses to generate Data Encryption Keys (DEKs).

FCS_CKM_EXT.3, Cryptographic Key Generation, requires the TSF to generate and manage the strength of Key Encryption Keys (KEKs).

FCS_CKM_EXT.5, TSF Wipe, requires the TSF to implement a cryptographic or other mechanism to make

TSF data unreadable.

FCS_CKM_EXT.6, Salt Generation, requires the TSF to generate salts in a specified manner.

FCS_CKM_EXT.7, Cryptographic Key Establishment, requires the TSF to perform key establishment regardless of whether the device is locked or unlocked.

FCS_CKM_EXT.8, Password-Based Key Derivation, requires that password-based key derivation be performed in accordance with specified standards.

Management: FCS_CKM_EXT.1

There are no management activities foreseen.

Audit: FCS_CKM_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Generation of a REK.

FCS_CKM_EXT.1 Cryptographic Key Support

Hierarchical to:	No other components.
Dependencies to:	FCS_RBG.1 Random Bit Generation

FCS_CKM_EXT.1.1

The TSF shall support [**assignment**: *description of REKs*]

FCS_CKM_EXT.1.2

Each REK shall be hardware-isolated from the OS on the TSF in runtime.

FCS_CKM_EXT.1.3

Each REK shall be generated by an RBG in accordance with FCS_RBG.1.

Management: FCS_CKM_EXT.2

There are no management activities foreseen.

Audit: FCS_CKM_EXT.2

There are no auditable events foreseen.

FCS_CKM_EXT.2 Cryptographic Key Random Generation

Hierarchical to:	No other components.
Dependencies to:	FCS_RBG.1 Random Bit Generation

FCS_CKM_EXT.2.1

All DEKs shall be [**assignment**: *generation mechanism*] with entropy corresponding to the security strength of AES key sizes of 256 bits.

Management: FCS_CKM_EXT.3

There are no management activities foreseen.

Audit: FCS_CKM_EXT.3

There are no auditable events foreseen.

FCS_CKM_EXT.3 Cryptographic Key Generation

Hierarchical to:	No other components.
Dependencies to:	FCS_CKM.1 Cryptographic Key Generation FCS_COP.1 Cryptographic Operation FCS_RBG.1 Random Bit Generation

FCS_CKM_EXT.3.1

The TSF shall use [**assignment**: *description of KEKs*].

FCS_CKM_EXT.3.2

The TSF shall generate all KEKs using one of the following methods:

- Derive the KEK from a Password Authentication Factor according to FCS_COP.1.1 and

[**selection**:

- *Generate the KEK using an RBG that meets this profile (as specified in FCS_RBG.1)*
- *Generate the KEK using a key generation scheme that meets this profile (as specified in FCS_CKM.1/AKG or FCS_CKM.1/SKG)*
- *Combine the KEK from other KEKs in a way that preserves the effective entropy of each factor by [**selection**: using an XOR operation, concatenating the keys and using a KDF (as described in SP 800-108), concatenating the keys and using a KDF (as described in SP 800-56C), encrypting one key with another]*

].

Management: FCS_CKM_EXT.5

The following actions could be considered for the management functions in FMT:

- TSF wipe of protected data.
- TSF wipe of enterprise data.

Audit: FCS_CKM_EXT.5

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Failure of the wipe.

FCS_CKM_EXT.5 TSF Wipe

Hierarchical to:	No other components.
Dependencies to:	FCS_CKM.6 Timing and Event of Cryptographic Key Destruction FCS_RBG.1 Random Bit Generation

FCS_CKM_EXT.5.1

The TSF shall wipe all protected data by [**selection**:

- *Cryptographically erasing the encrypted DEKs or the KEKs in non-volatile memory by following the requirements in FCS_CKM_EXT.6.2*
- *Overwriting all PD according to the following rules:*
  - *For EEPROM, the destruction shall be executed by a single direct overwrite consisting of a pseudo random pattern using the TSF's RBG (as specified in FCS_RBG.1, followed by a read-verify.*
  - *For flash memory, that is not wear-leveled, the destruction shall be executed [**selection**: by a single direct overwrite consisting of zeros followed by a read-verify, by a block erase that erases the reference to memory that stores data as well as the data itself].*
  - *For flash memory, that is wear-leveled, the destruction shall be executed [**selection**: by a single direct overwrite consisting of zeros, by a block erase].*
  - *For non-volatile memory other than EEPROM and flash, the destruction shall be executed by a single direct overwrite with a random pattern that is changed before each write.*

].

FCS_CKM_EXT.5.2

The TSF shall perform a power cycle on conclusion of the wipe procedure.

Management: FCS_CKM_EXT.6

There are no management activities foreseen.

Audit: FCS_CKM_EXT.6

There are no auditable events foreseen.

FCS_CKM_EXT.6 Salt Generation

Hierarchical to:	No other components.
Dependencies to:	FCS_RBG.1 Random Bit Generation

FCS_CKM_EXT.6.1

The TSF shall generate all salts using an RBG that meets FCS_RBG.1.

Management: FCS_CKM_EXT.7

There are no management functions foreseen.

Audit: FCS_CKM_EXT.7

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP, PP-Module, functional package or ST:

- minimal: Success and failure of the activity;
- basic: The object attribute(s), and object value(s) excluding any sensitive information.

FCS_CKM_EXT.7 Cryptographic Key Establishment

Hierarchical to: No other components.

Dependencies to:

[FDP_ITC.1 Import of User Data without Security Attributes, or
FDP_ITC.2 Import of User Data with Security Attributes, or
FCS_CKM.1 Cryptographic Key Generation, or
FCS_CKM.5 Cryptographic Key Derivation, or
FCS_CKM_EXT.8 Password-Based Key Derivation],
[FCS_CKM_EXT.7 Cryptographic Key Distribution, or
FCS_COP.1 Cryptographic Operation]
FCS_CKM.6 Timing and Event of Cryptographic Key Destruction
FCS_COP.1 Cryptographic Operation

FCS_CKM_EXT.7.1

The TSF shall derive shared cryptographic keys with input from multiple parties in accordance with specified cryptographic key agreement algorithms [**assignment**: *cryptographic algorithm*] and specified cryptographic parameters [**assignment**: *cryptographic parameters*] that meet the following: [**assignment**: *list of standards*].

Management: FCS_CKM_EXT.8

There are no management functions foreseen.

Audit: FCS_CKM_EXT.8

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP, PP-Module, functional package or ST:

- minimal: Success and failure of the activity;
- basic: The object attribute(s), and object value(s) excluding any sensitive information.

FCS_CKM_EXT.8 Password-Based Key Derivation

Hierarchical to: No other components.

Dependencies to:

[FCS_CKM.2 Cryptographic Key Distribution, or
FCS_COP.1 Cryptographic Operation]
FCS_CKM_EXT.7 Cryptographic Key Establishment],
FCS_CKM.6 Timing and Event of Cryptographic Key Destruction

FCS_CKM_EXT.8.1

The TSF shall perform password-based key derivation functions in accordance with a specified cryptographic algorithm [HMAC- [**selection**: *SHA-384*, *SHA-512*], with iteration count of [**assignment**: *number of iterations*] using a randomly generated salt of length [**assignment**: *equal to or greater than 128*] and output cryptographic key sizes [**selection**: *256*, *384*, *512*] bits that meet the following standard: [NIST SP 800-132 (Section 5.3) [PBKDF2]].

B.2.1.2 FCS_HTTPS_EXT HTTPS Protocol

Family Behavior

This family defines requirements for implementation of the HTTPS protocol.

Component Leveling

FCS_HTTPS_EXT.1, HTTPS Protocol, requires the TSF to implement the HTTPS protocol in accordance with the specified standard, using TLS, and notifying the application if invalid.

Management: FCS_HTTPS_EXT.1

The following actions could be considered for the management functions in FMT:

- Configuring whether to allow or disallow establishment of a trusted channel if the peer or server certificate is deemed invalid.

Audit: FCS_HTTPS_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Failure of the certificate validity check.

FCS_HTTPS_EXT.1 HTTPS Protocol

Hierarchical to: No other components.

Dependencies to: FCS_TLS_EXT.1 TLS Protocol
FIA_X509_EXT.1 X.509 Validation of Certificates

FCS_HTTPS_EXT.1.1

The TSF shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2

The TSF shall implement HTTPS using TLS as defined in [**assignment**: *specification that defines TLS implementation requirements*].

FCS_HTTPS_EXT.1.3

The TSF shall notify the application and [**assignment**: *a response*] if the peer certificate is deemed invalid.

B.2.1.3 FCS_IV_EXT Initialization Vector Generation

Family Behavior

This family defines requirements for initialization vector generation in support of key generation.

Component Leveling

FCS_IV_EXT.1, Initialization Vector Generation, requires the TSF to generate IVs in accordance with a set of approved modes.

Management: FCS_IV_EXT.1

There are no management activities foreseen.

Audit: FCS_IV_EXT.1

There are no auditable events foreseen.

FCS_IV_EXT.1 Initialization Vector Generation

Hierarchical to: No other components.

Dependencies to: No dependencies.

FCS_IV_EXT.1.1

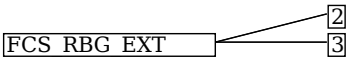
The TSF shall generate IVs in accordance with [**assignment**: *standard or specification for IV generation*].

B.2.1.4 FCS_RBG_EXT Random Bit Generation

Family Behavior

This family defines requirements for the generation of random bits.

Component Leveling



FCS_RBG_EXT.2, Random Bit Generator State Preservation, requires the TSF to save and restore the state of the RBG when powering off and starting up.

FCS_RBG_EXT.3, Support for Personalization String, requires the TSF to support a personalization string as a DRBG input parameter.

Management: FCS_RBG_EXT.2

There are no management activities foreseen.

Audit: FCS_RBG_EXT.2

There are no auditable events foreseen.

FCS_RBG_EXT.2 Random Bit Generator State Preservation

Hierarchical to: No other components.

Dependencies to: FCS_RBG.1 Random Bit Generation

FCS_RBG_EXT.2.1

The TSF shall save the state of the deterministic RBG at power-off, and shall use this state as input to the deterministic RBG at startup.

Management: FCS_RBG_EXT.3

There are no management activities foreseen.

Audit: FCS_RBG_EXT.3

There are no auditable events foreseen.

FCS_RBG_EXT.3 Support for Personalization String

Hierarchical to: No other components.

Dependencies to: FCS_RBG.1 Random Bit Generation

FCS_RBG_EXT.3.1

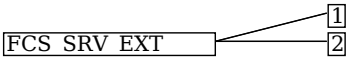
The TSF shall allow applications to add data to the deterministic RBG using the Personalization String as defined in SP 800-90A.

B.2.1.5 FCS_SRV_EXT Cryptographic Algorithm Services

Family Behavior

This family defines requirements for the ability of the TOE to make its cryptographic operations available to non-TSF components.

Component Leveling



FCS_SRV_EXT.1, Cryptographic Algorithm Services, requires the TSF to have externally-accessible cryptographic services for making algorithm functions available to applications.

FCS_SRV_EXT.2, Cryptographic Key Storage Services, requires the TSF to support its stored keys being usable by external applications through cryptographic algorithm services.

Management: FCS_SRV_EXT.1

There are no management activities foreseen.

Audit: FCS_SRV_EXT.1

There are no auditable events foreseen.

FCS_SRV_EXT.1 Cryptographic Algorithm Services

Hierarchical to: No other components.

Dependencies to: FCS_CKM.1 Cryptographic Key Generation
FCS_COP.1 Cryptographic Operation

FCS_SRV_EXT.1.1

The TSF shall provide a mechanism for applications to request the TSF to perform the following cryptographic operations: [**assignment**: *cryptographic operations defined by the TSF in FCS_CKM.1 or FCS_COP.1*]

Management: FCS_SRV_EXT.2

There are no management activities foreseen.

Audit: FCS_SRV_EXT.2

There are no auditable events foreseen.

FCS_SRV_EXT.2 Cryptographic Key Storage Services

Hierarchical to: No other components.

Dependencies to: FCS_COP.1 Cryptographic Operation

FCS_SRV_EXT.2.1

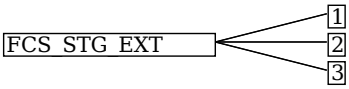
The TSF shall provide a mechanism for applications to request the TSF to perform the following cryptographic operations: [**assignment**: *cryptographic operations defined by the TSF in FCS_COP.1*] by keys stored in the secure key storage.

B.2.1.6 FCS_STG_EXT Cryptographic Key Storage

Family Behavior

This family defines requirements for the implementation of secure key storage with access control, confidentiality, and integrity protections.

Component Leveling



FCS_STG_EXT.1, Cryptographic Key Storage, requires the TSF to implement a secure key storage and defines the access restrictions to be enforced on this.

FCS_STG_EXT.2, Encrypted Cryptographic Key Storage, requires the TSF to implement confidentiality measures to protect the key storage.

FCS_STG_EXT.3, Integrity of Encrypted Key Storage, requires the TSF to implement integrity measures to protect the key storage.

Management: FCS_STG_EXT.1

The following actions could be considered for the management functions in FMT:

- Importing keys or secrets into the secure key storage.
- Destroying imported keys or secrets in the secure key storage.
- Approving exceptions for shared use of keys or secrets by multiple applications.
- Approving exceptions for destruction of keys or secrets by applications that did not import the key or secret

Audit: FCS_STG_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Import or destruction of key.
- Exceptions to use and destruction rules.

FCS_STG_EXT.1 Cryptographic Key Storage

Hierarchical to:	No other components.
Dependencies to:	[FCS_CKM.1 Cryptographic Key Generation, or FDP_ITC.1 Import of User Data without Security Attributes, or FDP_ITC.2 Import of User Data with Security Attributes] FMT_SMF.1 Specification of Management Functions FMT_SMR.1 Security Roles

FCS_STG_EXT.1.1

The TSF shall provide [**assignment**: *storage medium*] secure key storage for asymmetric private keys and [**assignment**: *list of secrets*].

FCS_STG_EXT.1.2

The TSF shall be capable of importing keys or secrets into the secure key storage upon request of [**assignment**: *list of users*] and [**assignment**: *list of other subjects*].

FCS_STG_EXT.1.3

The TSF shall be capable of destroying keys or secrets in the secure key storage upon request of [**selection**: *the user*, *the administrator*].

FCS_STG_EXT.1.4

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

FCS_STG_EXT.1.5

The TSF shall allow only the application that imported the key or secret to request that the key or secret be destroyed. Exceptions may only be explicitly authorized by [**assignment**: *list of subjects*].

Management: FCS_STG_EXT.2

There are no management activities foreseen.

Audit: FCS_STG_EXT.2

There are no auditable events foreseen.

FCS_STG_EXT.2 Encrypted Cryptographic Key Storage

Hierarchical to:	No other components.

Dependencies to: FCS_CKM_EXT.3 Cryptographic Key Generation
FCS_COP.1 Cryptographic Operation
FCS_STG_EXT.1 Cryptographic Key Storage

FCS_STG_EXT.2.1

The TSF shall encrypt all DEKs, KEKs, [**assignment**: *any long-term trusted channel key material*] and [**assignment**: *other secrets*] by KEKs that are [**assignment**: *protection mechanism*].

FCS_STG_EXT.2.2

DEKs, KEKs, [**assignment**: *any long-term trusted channel key material*] and [**selection**: *all software-based key storage*, *no other keys*] shall be encrypted using one of the following methods: [**selection**:

- *using a SP800-56B key establishment scheme*
- *using AES in the [**selection**: Key Wrap (KW) mode, Key Wrap with Padding (KWP) mode, GCM, CCM, CBC mode]*

].

Management: FCS_STG_EXT.3

There are no management activities foreseen.

Audit: FCS_STG_EXT.3

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Failure to verify the integrity of stored key.

FCS_STG_EXT.3 Integrity of Encrypted Key Storage

Hierarchical to: No other components.

Dependencies to: FCS_COP.1 Cryptographic Operation
FCS_STG_EXT.2 Encrypted Cryptographic Key Storage

FCS_STG_EXT.3.1

The TSF shall protect the integrity of any encrypted DEKs and KEKs and [**selection**: *long-term trusted channel key material*, *all software-based key storage*, *no other keys*] by [**selection**:

- *[**selection**: GCM, CCM, Key Wrap, Key Wrap with Padding] cipher mode for encryption according to FCS_STG_EXT.2*
- *a hash (FCS_COP.1) of the stored key that is encrypted by a key protected by FCS_STG_EXT.2*
- *a keyed hash (FCS_COP.1) using a key protected by a key protected by FCS_STG_EXT.2*
- *a digital signature of the stored key using an asymmetric key protected according to FCS_STG_EXT.2*
- *an immediate application of the key for decrypting the protected data followed by a successful verification of the decrypted data with previously known information*

].

FCS_STG_EXT.3.2

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

B.2.2 Class: Identification and Authentication (FIA)

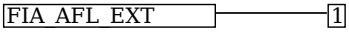
This PP defines the following extended components as part of the FIA class originally defined by CC Part 2:

B.2.2.1 FIA_AFL_EXT Authentication Failures

Family Behavior

This family defines requirements for authentication failure handling that are not addressed by the FIA_AFL family in CC Part 2.

Component Leveling



FIA_AFL_EXT.1, Authentication Failure Handling, requires the TSF be able to manage unsuccessful authentication attempts and limit the number of attempts for each method.

Management: FIA_AFL_EXT.1

The following actions could be considered for the management functions in FMT:

- Configuration of authentication failure limit.

Audit: FIA_AFL_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Exceeding configured authentication failure limit.

FIA_AFL_EXT.1 Authentication Failure Handling

Hierarchical to:	No other components.
Dependencies to:	FCS_CKM_EXT.5 TSF Wipe FIA_UAU.1 Timing of Authentication FIA_UAU.5 Multiple Authentication Mechanisms

FIA_AFL_EXT.1.1

The TSF shall consider password and [**assignment**: *list of acceptable authentication mechanisms*] as critical authentication mechanisms.

FIA_AFL_EXT.1.2

The TSF shall detect when a configurable positive integer within [**assignment**: *range of acceptable values for each authentication mechanism*] of [**selection**: *unique*, *non-unique*] unsuccessful authentication attempts occur related to last successful authentication for each authentication mechanism.

FIA_AFL_EXT.1.3

The TSF shall maintain the number of unsuccessful authentication attempts that have occurred upon power off.

FIA_AFL_EXT.1.4

When the defined number of unsuccessful authentication attempts has exceeded the maximum allowed for a given authentication mechanism, all future authentication attempts will be limited to other available authentication mechanisms, unless the given mechanism is designated as a critical authentication mechanism.

FIA_AFL_EXT.1.5

When the defined number of unsuccessful authentication attempts for the last available authentication mechanism or single critical authentication mechanism has been surpassed, the TSF shall perform a wipe of all protected data.

FIA_AFL_EXT.1.6

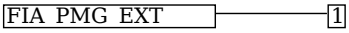
The TSF shall increment the number of unsuccessful authentication attempts prior to notifying the user that the authentication was unsuccessful.

B.2.2.2 FIA_PMG_EXT Password Management

Family Behavior

This family defines requirements for the composition of password credentials.

Component Leveling



FIA_PMG_EXT.1, Password Management, requires the TSF to enforce character length and composition requirements for password credentials.

Management: FIA_PMG_EXT.1

The following actions could be considered for the management functions in FMT:

- Configuring password policy.

Audit: FIA_PMG_EXT.1

There are no auditable events foreseen.

FIA_PMG_EXT.1 Password Management

Hierarchical to:	No other components.
Dependencies to:	FIA_UAU.5 Multiple Authentication Mechanisms

FIA_PMG_EXT.1.1

The TSF shall support the following for the Password Authentication Factor:

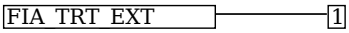
1. Passwords shall be able to be composed of any combination of [**selection**: *upper and lower case letters*, [***assignment**: a character set of at least 52 characters*]], numbers, and special characters: [**selection**: "*!*", "@", "#", "*$*", "%", "^", "&", "*", "(", ")", [***assignment**: other characters*]] ;
2. Password length up to [**assignment**: *an integer greater than or equal to 14*] characters shall be supported.

B.2.2.3 FIA_TRT_EXT Authentication Throttling

Family Behavior

This family defines requirements for prevention of brute-force authentication attempts.

Component Leveling



FIA_TRT_EXT.1, Authentication Throttling, requires the TSF to limit authentication attempts by number of attempts in a set amount of time.

Management: FIA_TRT_EXT.1

There are no management activities foreseen.

Audit: FIA_TRT_EXT.1

There are no auditable events foreseen.

FIA_TRT_EXT.1 Authentication Throttling

Hierarchical to: No other components.

Dependencies to: FIA_UAU.5 Multiple Authentication Mechanisms

FIA_TRT_EXT.1.1

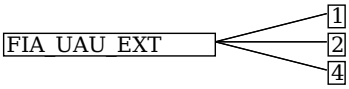
The TSF shall limit automated user authentication attempts by [**selection**: *preventing authentication via an external port*, *enforcing a delay between incorrect authentication attempts*] for all authentication mechanisms selected in FIA_UAU.5.1. The minimum delay shall be such that no more than 10 attempts can be attempted per 500 milliseconds.

B.2.2.4 FIA_UAU_EXT User Authentication

Family Behavior

This family defines requirements for user authentication that are not addressed by FIA_UAU in CC Part 2.

Component Leveling



FIA_UAU_EXT.1, Authentication for Cryptographic Operation, requires the TSF enforce data-at-rest protection until successful authentication has occurred.

FIA_UAU_EXT.2, Timing of Authentication, requires the TSF to prevent a subject's use of TOE until the user is authenticated.

FIA_UAU_EXT.4, Secondary User Authentication, requires the TSF to enforce the use of a secondary authentication factor to access certain user data.

Management: FIA_UAU_EXT.1

There are no management activities foreseen.

Audit: FIA_UAU_EXT.1

There are no auditable events foreseen.

FIA_UAU_EXT.1 Authentication for Cryptographic Operation

Hierarchical to: No other components.

Dependencies to: FDP_DAR_EXT.1 Protected Data Encryption
FDP_DAR_EXT.2 Sensitive Data Encryption

FIA_UAU_EXT.1.1

The TSF shall require the user to present the Password Authentication Factor prior to decryption of protected data and encrypted DEKs, KEKs and [**selection**: *long-term trusted channel key material*, *all software-based key storage*, *no other keys*] at startup.

Management: FIA_UAU_EXT.2

The following actions could be considered for the management functions in FMT:

- Enabling/disabling display TSF notifications while in the locked state.
- Enabling/disabling bypass of local user authentication.

Audit: FIA_UAU_EXT.2

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Action performed before authentication.

FIA_UAU_EXT.2 Timing of Authentication

Hierarchical to: No other components.

Dependencies to: No dependencies.

FIA_UAU_EXT.2.1

The TSF shall allow [**selection**: ***[assignment**: list of actions] , no actions*] on behalf of the user to be performed before the user is authenticated.

FIA_UAU_EXT.2.2

The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

Management: FIA_UAU_EXT.4

There are no management activities foreseen.

Audit: FIA_UAU_EXT.4

There are no auditable events foreseen.

FIA_UAU_EXT.4 Secondary User Authentication

Hierarchical to: No other components.

Dependencies to: FDP_ACF_EXT.2 Access Control for System Resources
FIA_UAU.5 Multiple Authentication Mechanisms

FIA_UAU_EXT.4.1

The TSF shall provide a secondary authentication mechanism for accessing Enterprise applications and resources. The secondary authentication mechanism shall control access to the Enterprise application and shared resources and shall be incorporated into the encryption of protected and sensitive data belonging to Enterprise applications and shared resources.

FIA_UAU_EXT.4.2

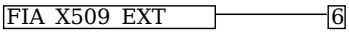
The TSF shall require the user to present the secondary authentication factor prior to decryption of Enterprise application data and Enterprise shared resource data.

B.2.2.5 FIA_X509_EXT X.509 Certificates

Family Behavior

This family defines requirements for the management and use of X.509 certificates.

Component Leveling



FIA_X509_EXT.6, Request Validation of Certificates, requires the TSF to make a certificate validation service available to environmental components.

Management: FIA_X509_EXT.6

There are no management activities foreseen.

Audit: FIA_X509_EXT.6

There are no auditable events foreseen.

FIA_X509_EXT.6 Request Validation of Certificates

Hierarchical to: No other components.

Dependencies to: FIA_X509_EXT.1 X.509 Validation of Certificates

FIA_X509_EXT.6.1

The TSF shall provide a certificate validation service to applications.

FIA_X509_EXT.6.2

The TSF shall respond to the requesting application with the success or failure of the validation.

B.2.3 Class: Protection of the TSF (FPT)

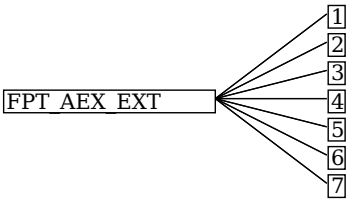
This PP defines the following extended components as part of the FPT class originally defined by CC Part 2:

B.2.3.1 FPT_AEX_EXT Anti-Exploitation Capabilities

Family Behavior

This family defines requirements for protecting against common types of software exploitation techniques.

Component Leveling



FPT_AEX_EXT.1, Application Address Space Layout Randomization, requires the TSF to support address space layout randomization (ASLR).

FPT_AEX_EXT.2, Memory Page Permissions, requires the TSF to enforce access permissions on physical memory.

FPT_AEX_EXT.3, Stack Overflow Protection, requires the TSF to implement stack overflow protection.

FPT_AEX_EXT.4, Domain Isolation, requires the TSF to protect itself from untrusted subjects and enforce address space isolation.

FPT_AEX_EXT.5, Kernel Address Space Layout Randomization, requires the TSF to provide ASLR to the kernel.

FPT_AEX_EXT.6, Write or Execute Memory Page Permissions, requires the TSF to prevent physical memory from being both writable and executable.

FPT_AEX_EXT.7, Heap Overflow Protection, requires the TSF to support heap-based buffer overflow protection.

Management: FPT_AEX_EXT.1

There are no management activities foreseen.

Audit: FPT_AEX_EXT.1

There are no auditable events foreseen.

FPT_AEX_EXT.1 Application Address Space Layout Randomization

Hierarchical to: No other components.

Dependencies to: FCS_RBG.1 Random Bit Generation

FPT_AEX_EXT.1.1

The TSF shall provide address space layout randomization ASLR to applications.

FPT_AEX_EXT.1.2

The base address of any user-space memory mapping will consist of at least 8 unpredictable bits.

Management: FPT_AEX_EXT.2

There are no management activities foreseen.

Audit: FPT_AEX_EXT.2

There are no auditable events foreseen.

FPT_AEX_EXT.2 Memory Page Permissions

Hierarchical to: No other components.

Dependencies to: No dependencies.

FPT_AEX_EXT.2.1

The TSF shall be able to enforce read, write, and execute permissions on every page of physical memory.

Management: FPT_AEX_EXT.3

There are no management activities foreseen.

Audit: FPT_AEX_EXT.3

There are no auditable events foreseen.

FPT_AEX_EXT.3 Stack Overflow Protection

Hierarchical to: No other components.

Dependencies to: No dependencies.

FPT_AEX_EXT.3.1

TSF processes that execute in a non-privileged execution domain on the application processor shall implement stack-based buffer overflow protection.

Management: FPT_AEX_EXT.4

There are no management activities foreseen.

Audit: FPT_AEX_EXT.4

There are no auditable events foreseen.

FPT_AEX_EXT.4 Domain Isolation

Hierarchical to: No other components.

Dependencies to: No dependencies.

FPT_AEX_EXT.4.1

The TSF shall protect itself from modification by untrusted subjects.

FPT_AEX_EXT.4.2

The TSF shall enforce isolation of address space between applications.

Management: FPT_AEX_EXT.5

There are no management activities foreseen.

Audit: FPT_AEX_EXT.5

There are no auditable events foreseen.

FPT_AEX_EXT.5 Kernel Address Space Layout Randomization

Hierarchical to: No other components.

Dependencies to: FCS_RBG.1 Random Bit Generation (RBG)

FPT_AEX_EXT.5.1

The TSF shall provide address space layout randomization (ASLR) to the kernel.

FPT_AEX_EXT.5.2

The base address of any kernel-space memory mapping will consist of [**assignment**: *number greater than or equal to 4*] unpredictable bits.

Management: FPT_AEX_EXT.6

There are no management activities foreseen.

Audit: FPT_AEX_EXT.6

There are no auditable events foreseen.

FPT_AEX_EXT.6 Write or Execute Memory Page Permissions

Hierarchical to: No other components.

Dependencies to: No dependencies.

FPT_AEX_EXT.6.1

The TSF shall prevent write and execute permissions from being simultaneously granted to any page of physical memory [**selection**: *with no exceptions, [**assignment**: specific exceptions]*].

Management: FPT_AEX_EXT.7

There are no management activities foreseen.

Audit: FPT_AEX_EXT.7

There are no auditable events foreseen.

FPT_AEX_EXT.7 Heap Overflow Protection

Hierarchical to: No other components.

Dependencies to: No dependencies.

FPT_AEX_EXT.7.1

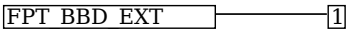
The TSF shall include heap-based buffer overflow protections in the runtime environment it provides to processes that execute on the application processor.

B.2.3.2 FPT_BBD_EXT Baseband Processing

Family Behavior

This family defines requirements for separation of baseband and application processor execution.

Component Leveling



FPT_BBD_EXT.1, Application Processor Mediation, requires the TSF to enforce separation between baseband and application processor execution except through application processor mechanisms.

Management: FPT_BBD_EXT.1

There are no management activities foreseen.

Audit: FPT_BBD_EXT.1

There are no auditable events foreseen.

FPT_BBD_EXT.1 Application Processor Mediation

Hierarchical to: No other components.

Dependencies to: No dependencies.

FPT_BBD_EXT.1.1

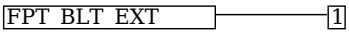
The TSF shall prevent code executing on any baseband processor (BP) from accessing application processor (AP) resources except when mediated by the AP.

B.2.3.3 FPT_BLT_EXT Limitation of Bluetooth Profile Support

Family Behavior

This family defines requirements for limiting Bluetooth capabilities without user action.

Component Leveling



FPT_BLT_EXT.1, Limitation of Bluetooth Profile Support, requires the TSF to maintain a disabled by default posture for Bluetooth profiles.

Management: FPT_BLT_EXT.1

There are no management activities foreseen.

Audit: FPT_BLT_EXT.1

There are no auditable events foreseen.

FPT_BLT_EXT.1 Limitation of Bluetooth Profile Support

Hierarchical to: No other components.

Dependencies to: No dependencies.

FPT_BLT_EXT.1.1

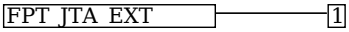
The TSF shall disable support for [**assignment**: *list of Bluetooth profiles*] Bluetooth profiles when they are not currently being used by an application on the Mobile Device, and shall require explicit user action to enable them.

B.2.3.4 FPT_JTA_EXT JTAG Disablement

Family Behavior

This family defines requirements for JTAG interface access limitations.

Component Leveling



FPT_JTA_EXT.1, JTAG Disablement, requires the TSF to specify the mechanism used to restrict access to its JTAG interface.

Management: FPT_JTA_EXT.1

There are no management activities foreseen.

Audit: FPT_JTA_EXT.1

There are no auditable events foreseen.

FPT_JTA_EXT.1 JTAG Disablement

Hierarchical to: No other components.

Dependencies to: No dependencies.

FPT_JTA_EXT.1.1

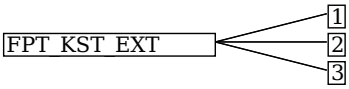
The TSF shall [**assignment**: *list access control mechanisms*] to JTAG.

B.2.3.5 FPT_KST_EXT Key Storage

Family Behavior

This family defines requirements for protecting plaintext keys.

Component Leveling



FPT_KST_EXT.1, Key Storage, requires the TSF to avoid storage of plaintext keys in readable memory.

FPT_KST_EXT.2, No Key Transmission, requires the TSF to prevent transmitting plaintext key material to the operational environment.

FPT_KST_EXT.3, No Plaintext Key Export, requires the TSF to prevent the export of plaintext keys.

Management: FPT_KST_EXT.1

There are no management activities foreseen.

Audit: FPT_KST_EXT.1

There are no auditable events foreseen.

FPT_KST_EXT.1 Key Storage

Hierarchical to: No other components.

Dependencies to: No dependencies.

FPT_KST_EXT.1.1

The TSF shall not store any plaintext key material in readable non-volatile memory.

Management: FPT_KST_EXT.2

There are no management activities foreseen.

Audit: FPT_KST_EXT.2

There are no auditable events foreseen.

FPT_KST_EXT.2 No Key Transmission

Hierarchical to: No other components.

Dependencies to: No dependencies.

FPT_KST_EXT.2.1

The TSF shall not transmit any plaintext key material outside the security boundary of the TOE.

Management: FPT_KST_EXT.3

There are no management activities foreseen.

Audit: FPT_KST_EXT.3

There are no auditable events foreseen.

FPT_KST_EXT.3 No Plaintext Key Export

Hierarchical to:	No other components.
Dependencies to:	No dependencies.

FPT_KST_EXT.3.1

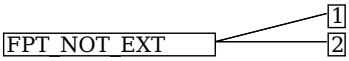
The TSF shall ensure it is not possible for the TOE users to export plaintext keys.

B.2.3.6 FPT_NOT_EXT Self-Test Notification

Family Behavior

This family defines requirements for generation of notifications in response to completed self-tests.

Component Leveling



FPT_NOT_EXT.1, Self-Test Notification, requires the TSF to become non-operational when certain failures occur.

FPT_NOT_EXT.2, Software Integrity Verification, requires the TSF to generate and sign software integrity verification values.

Management: FPT_NOT_EXT.1

There are no management activities foreseen.

Audit: FPT_NOT_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Measurement of TSF software.

FPT_NOT_EXT.1 Self-Test Notification

Hierarchical to:	No other components.
Dependencies to:	FPT_TST_EXT.1 TSF Cryptographic Functionality Testing FPT_TST_EXT.2 TSF Integrity Checking

FPT_NOT_EXT.1.1

The TSF shall transition to non-operational mode and [**selection**: *log failures in the audit record*, *notify the administrator*, *[**assignment**: other actions]*, *no other actions*] when the following types of failures occur:

- failures of the self-tests
- TSF software integrity verification failures
- [**selection**: *no other failures*, *[**assignment**: other failures]*]

Management: FPT_NOT_EXT.2

The following actions could be considered for the management functions in FMT:

- Retrieval of TSF software integrity verification values.

Audit: FPT_NOT_EXT.2

There are no auditable events foreseen.

FPT_NOT_EXT.2 Software Integrity Verification

Hierarchical to:	No other components.
Dependencies to:	FCS_COP.1 Cryptographic Operation

FPT_NOT_EXT.2.1

The TSF shall [**selection**: *audit*, *provide the administrator with*] TSF-software integrity verification values.

FPT_NOT_EXT.2.2

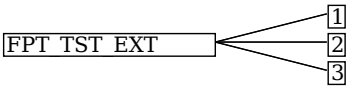
The TSF shall cryptographically sign all integrity verification values.

B.2.3.7 FPT_TST_EXT TSF Self Test

Family Behavior

This family defines requirements for execution of self-tests that are not addressed by FPT_TST in CC Part 2.

Component Leveling



FPT_TST_EXT.1, TSF Cryptographic Functionality Testing, requires the TSF to run self-test at start-up to verify correct operation.

FPT_TST_EXT.2, TSF Integrity Checking, requires the TSF to verify code and bootchain integrity is stored prior to execution by a designated key or hash.

FPT_TST_EXT.3, TSF Integrity Testing, requires the TSF to validate a code signing certificate before the associated code is executed.

Management: FPT_TST_EXT.1

There are no management activities foreseen.

Audit: FPT_TST_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Initiation of self-test.
- Failure of self-test.

FPT_TST_EXT.1 TSF Cryptographic Functionality Testing

Hierarchical to:	No other components.
Dependencies to:	FCS_COP.1 Cryptographic Operation

FPT_TST_EXT.1.1

The TSF shall run a suite of self-tests during initial start-up (on power on) to demonstrate the correct operation of all cryptographic functionality.

Management: FPT_TST_EXT.2

There are no management activities foreseen.

Audit: FPT_TST_EXT.2

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Start-up of TOE.
- [**selection**: *Detected integrity violation*, *None*]

FPT_TST_EXT.2 TSF Integrity Checking

Hierarchical to:	No other components.
Dependencies to:	FCS_COP.1 Cryptographic Operation

FPT_TST_EXT.2.1

The TSF shall verify the integrity of [**assignment**: *TSF data*] stored in mutable media prior to its execution through the use of [**assignment**: *cryptographic or immutable hardware mechanism*].

Management: FPT_TST_EXT.3

There are no management activities foreseen.

Audit: FPT_TST_EXT.3

There are no auditable events foreseen.

FPT_TST_EXT.3 TSF Integrity Testing

Hierarchical to:	No other components.
Dependencies to:	FPT_TST_EXT.2 TSF Integrity Checking FIA_X509_EXT.1 X.509 Validation of Certificates FIA_X509_EXT.2 X.509 Certificate Authentication

FPT_TST_EXT.3.1

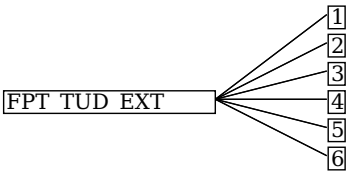
The TSF shall not execute code if the code signing certificate is deemed invalid.

B.2.3.8 FPT_TUD_EXT TSF Updates

Family Behavior

This family defines requirements for trusted updates.

Component Leveling



FPT_TUD_EXT.1, TSF Version Query, requires the TSF to provide authorized users the ability to query the version of the TOE hardware, TOE software, and installed applications.

FPT_TUD_EXT.2, TSF Update Verification, requires the TSF to ensure that system software updates are digitally signed prior to installation.

FPT_TUD_EXT.3, Application Signing, requires the TSF to ensure that application software updates are digitally signed prior to installation.

FPT_TUD_EXT.4, Trusted Update Verification, requires the TSF to enforce validity of system software's code signing certificate prior to installation.

FPT_TUD_EXT.5, Application Verification, requires the TSF to enforce validity of application software's code signing certificate prior to installation.

FPT_TUD_EXT.6, Trusted Update Verification, requires the TSF to prevent the intentional rollback of software updates.

Management: FPT_TUD_EXT.1

There are no management activities foreseen.

Audit: FPT_TUD_EXT.1

There are no auditable events foreseen.

FPT_TUD_EXT.1 TSF Version Query

Hierarchical to: No other components.

Dependencies to: FMT_SMR.1 Security Roles

FPT_TUD_EXT.1.1

The TSF shall provide authorized users the ability to query the current version of the TOE firmware/software.

FPT_TUD_EXT.1.2

The TSF shall provide authorized users the ability to query the current version of the hardware model of the device.

FPT_TUD_EXT.1.3

The TSF shall provide authorized users the ability to query the current version of installed mobile applications.

Management: FPT_TUD_EXT.2

The following actions could be considered for the management functions in FMT:

- Updating of system software.

Audit: FPT_TUD_EXT.2

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Success or failure of signature verification for applications.

FPT_TUD_EXT.2 TSF Update Verification

Hierarchical to: No other components.

Dependencies to: FCS_COP.1 Cryptographic Operation

FPT_TUD_EXT.2.1

The TSF shall verify software updates to the Application Processor system software and [**selection**: *[**assignment**: other processor system software] , no other processor system software*] using a digital signature verified by the manufacturer trusted key prior to installing those updates.

FPT_TUD_EXT.2.2

The TSF shall [**selection**: *never update, update only by verified software*] the TSF boot integrity [**selection**: *key, hash*].

FPT_TUD_EXT.2.3

The TSF shall verify that the digital signature verification key used for TSF updates [**selection**: *is validated to a public key in the Trust Anchor Database, matches an immutable hardware public key*].

Management: FPT_TUD_EXT.3

There are no management activities foreseen.

Audit: FPT_TUD_EXT.3

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Success or failure of signature verification for applications.

FPT_TUD_EXT.3 Application Signing

Hierarchical to:	No other components.
Dependencies to:	FCS_COP.1 Cryptographic Operation

FPT_TUD_EXT.3.1

The TSF shall verify mobile application software using a digital signature mechanism prior to installation.

Management: FPT_TUD_EXT.4

There are no management activities foreseen.

Audit: FPT_TUD_EXT.4

There are no auditable events foreseen.

FPT_TUD_EXT.4 Trusted Update Verification

Hierarchical to:	No other components.
Dependencies to:	FIA_X509_EXT.1 X.509 Validation of Certificates FIA_X509_EXT.2 X.509 Certificate Authentication

FPT_TUD_EXT.4.1

The TSF shall not install code if the code signing certificate is deemed invalid.

Management: FPT_TUD_EXT.5

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Configure certificate or public key used to validate digital signature on applications.

Audit: FPT_TUD_EXT.5

There are no auditable events foreseen.

FPT_TUD_EXT.5 Application Verification

Hierarchical to:	No other components.
Dependencies to:	FIA_X509_EXT.1 X.509 Validation of Certificates FIA_X509_EXT.2 X.509 Certificate Authentication

FPT_TUD_EXT.5.1

The TSF shall by default only install mobile applications cryptographically verified by [**selection**: *a built-in X.509v3 certificate, a configured X.509v3 certificate*].

Management: FPT_TUD_EXT.6

There are no management activities foreseen.

Audit: FPT_TUD_EXT.6

There are no auditable events foreseen.

FPT_TUD_EXT.6 Trusted Update Verification

Hierarchical to: No other components.

Dependencies to: No dependencies.

FPT_TUD_EXT.6.1

The TSF shall verify that software updates to the TSF are a current or later version than the current version of the TSF.

B.2.4 Class: Security Management (FMT)

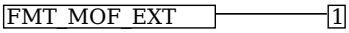
This PP defines the following extended components as part of the FMT class originally defined by CC Part 2:

B.2.4.1 FMT_MOF_EXT Management of Functions in TSF

Family Behavior

This family defines requirements for authorization to manage the behavior of the TSF that are not addressed by FMT_MOF in CC Part 2.

Component Leveling



FMT_MOF_EXT.1, Management of Security Functions Behavior, requires the TSF to apply restrictions to access its management functions to the authorized roles.

Management: FMT_MOF_EXT.1

The following actions could be considered for the management functions in FMT:

- Managing the group of roles that can interact with the functions in the TSF.

Audit: FMT_MOF_EXT.1

There are no auditable events foreseen.

FMT_MOF_EXT.1 Management of Security Functions Behavior

Hierarchical to: No other components.

Dependencies to: FMT_SMF.1 Specification of Management Functions
FMT_SMR.1 Security Roles

FMT_MOF_EXT.1.1

The TSF shall restrict the ability to perform the functions [**assignment**: *reference to list of management functions*] to the user.

FMT_MOF_EXT.1.2

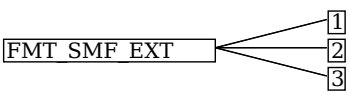
The TSF shall restrict the ability to perform the functions [**assignment**: *reference to list of management functions*] to the administrator when the device is enrolled and according to the administrator-configured policy.

B.2.4.2 FMT_SMF_EXT Specification of Management Functions

Family Behavior

This family defines requirements for security-relevant management functions that are not addressed by FMT_SMF in CC Part 2.

Component Leveling



FMT_SMF_EXT.1, Specification of Management Functions, requires the TSF to define the management functions that it implements.

FMT_SMF_EXT.2, Specification of Remediation Actions, requires the TSF to automatically perform specific management functions in response to a specific event.

FMT_SMF_EXT.3, Current Administrator, requires the TSF to provide users with a list of administrators and their specified functions.

Management: FMT_SMF_EXT.1

There are no management activities foreseen.

Audit: FMT_SMF_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Initiation of policy update.
- Change of settings.
- Success or failure of function
- Initiation of software update.
- Initiation of application installation or update.

FMT_SMF_EXT.1 Specification of Management Functions

Hierarchical to: No other components.

Dependencies to: No dependencies.

FMT_SMF_EXT.1.1

The TSF shall be capable of performing [**assignment**: *list of management functions*].

Management: FMT_SMF_EXT.2

The following actions could be considered for the management functions in FMT:

- Configuration of the functions that are performed in response to unenrollment event.

Audit: FMT_SMF_EXT.2

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Initiation of unenrollment.
- Completion of unenrollment.

FMT_SMF_EXT.2 Specification of Remediation Actions

Hierarchical to: No other components.

Dependencies to: No dependencies.

FMT_SMF_EXT.2.1

The TSF shall offer [**assignment**: *list of remediation actions*] upon unenrollment and [**assignment**: *list of triggers*].

Management: FMT_SMF_EXT.3

There are no management activities foreseen.

Audit: FMT_SMF_EXT.3

There are no auditable events foreseen.

FMT_SMF_EXT.3 Current Administrator

Hierarchical to: No other components.

Dependencies to: FMT_SMR.1 Security Roles

FMT_SMF_EXT.3.1

The TSF shall provide a mechanism that allows users to view a list of currently authorized administrators and the management functions that each administrator is authorized to perform.

B.2.5 Class: TOE Access (FTA)

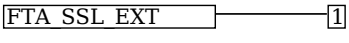
This PP defines the following extended components as part of the FTA class originally defined by CC Part 2:

B.2.5.1 FTA_SSL_EXT Session Locking and Termination

Family Behavior

This family defines requirements for session locking capabilities that are not addressed by FTA_SSL in CC Part 2.

Component Leveling



FTA_SSL_EXT.1, TSF- and User-initiated Locked State , requires the TSF to manage the transition to a locked state and what operations can be performed.

Management: FTA_SSL_EXT.1

The following actions could be considered for the management functions in FMT:

- Configuring session locking policy.
- Transitioning to the locked state.

Audit: FTA_SSL_EXT.1

There are no auditable events foreseen.

FTA_SSL_EXT.1 TSF- and User-initiated Locked State

Hierarchical to:	No other components.
Dependencies to:	FMT_SMR.1 Security Roles FPT_STM.1 Reliable Time Stamps

FTA_SSL_EXT.1.1

The TSF shall transition to a locked state after a time interval of inactivity.

FTA_SSL_EXT.1.2

The TSF shall transition to a locked state after initiation by either the user or the administrator.

FTA_SSL_EXT.1.3

The TSF shall, upon transitioning to the locked state, perform the following operations:

- Clearing or overwriting display devices, obscuring the previous contents;
- [**assignment**: *Other actions performed upon transitioning to the locked state*].

B.2.6 Class: Trusted Path/Channels (FTP)

This PP defines the following extended components as part of the FTP class originally defined by CC Part 2:

B.2.6.1 FTP_ITC_EXT Inter-TSF Trusted Channel

Family Behavior

This family defines requirements for trusted channels that are not addressed by FTP_ITC in CC Part 2 because they apply specifically to channels required by a mobile device.

Component Leveling

FTP_ITC_EXT.1, Trusted Channel Communication, requires the TSF to manage the communication channel between itself and other trusted products.

Management: FTP_ITC_EXT.1

The following actions could be considered for the management functions in FMT:

- Configuring the actions that require trusted channel, if applicable.
- Enabling/disabling communications protocols where the TSF acts as a server.

Audit: FTP_ITC_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Initiation and termination of trusted channel.

FTP_ITC_EXT.1 Trusted Channel Communication

Hierarchical to:	No other components.
Dependencies to:	No dependencies.

FTP_ITC_EXT.1.1

The TSF shall use

- 802.11-2012 in accordance with [**assignment**: *requirements or standards defining implementation of this protocol*],
- 802.1X in accordance with [**assignment**: *requirements or standards defining implementation of this protocol*],
- EAP-TLS in accordance with [**assignment**: *requirements or standards defining implementation of this protocol*],
- Mutually authenticated TLS in accordance with [**assignment**: *requirements or standards defining implementation of this protocol*]

and [**assignment**: *other protocols*] protocols to provide a communication channel between itself and another trusted IT product using certificates as defined in [**assignment**: *requirement or standard defining the use of certificates*] that is logically distinct from other communication channels, provides assured identification of its end points, protects channel data from disclosure, and detects modification of

the channel data.

FTP_ITC_EXT.1.2

The TSF shall permit the TSF to initiate communication via the trusted channel.

FTP_ITC_EXT.1.3

The TSF shall initiate communication via the trusted channel for wireless access point connections, administrative communication, configured enterprise connections, and [**selection**: *OTA updates*, *no other connections*].

B.2.7 Class: User Data Protection (FDP)

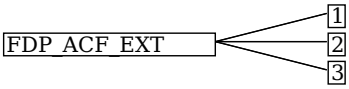
This PP defines the following extended components as part of the FDP class originally defined by CC Part 2:

B.2.7.1 FDP_ACF_EXT Access Control Functions

Family Behavior

This family defines the rules for access control functions that are not addressed by the FDP_ACF family in CC Part 2.

Component Leveling



FDP_ACF_EXT.1, Access Control for System Services, requires the TSF to be able to control access to its own services.

FDP_ACF_EXT.2, Access Control for System Resources, requires the TSF to be able to provide separate copies of system resources for different application groups.

FDP_ACF_EXT.3, Security Attribute Based Access Control, requires the TSF to enforce policies on applications that prohibit write and execute permissions from being granted simultaneously.

Management: FDP_ACF_EXT.1

The following actions could be considered for the management functions in FMT:

- Placing applications into application groups based on enterprise configuration settings.
- Enabling/disabling location services.
- Enabling/disabling data signaling over externally-accessible hardware ports.

Audit: FDP_ACF_EXT.1

There are no auditable events foreseen.

FDP_ACF_EXT.1 Access Control for System Services

Hierarchical to:	No other components.
Dependencies to:	FMT_SMR.1 Security Roles

FDP_ACF_EXT.1.1

The TSF shall provide a mechanism to restrict the system services that are accessible to an application.

FDP_ACF_EXT.1.2

The TSF shall provide an access control policy that prevents [**assignment**: *list of subjects*] from accessing [**selection**: *all*, *private*] data stored by other [**assignment**: *list of subjects*]. Exceptions may only be explicitly authorized for such sharing by [**assignment**: *list of authorized subjects*].

Management: FDP_ACF_EXT.2

The following actions could be considered for the management functions in FMT:

- Approving exceptions for sharing data between applications or groups of applications.

Audit: FDP_ACF_EXT.2

There are no auditable events foreseen.

FDP_ACF_EXT.2 Access Control for System Resources

Hierarchical to:	No other components.
Dependencies to:	FDP_ACF_EXT.1 Access Control for System Services FMT_SMR.1 Security Roles

FDP_ACF_EXT.2.1

The TSF shall provide a separate [**selection**: *address book*, *calendar*, *keystore*, *account credential database*, [***assignment**: list of additional resources]*] for each application group and only allow applications within that process group to access the resource. Exceptions may only be explicitly authorized for such sharing by [**selection**: *the user*, *the administrator*, *no one*].

Management: FDP_ACF_EXT.3

There are no management activities foreseen.

Audit: FDP_ACF_EXT.3

There are no auditable events foreseen.

FDP_ACF_EXT.3 Security Attribute Based Access Control

Hierarchical to: No other components.

Dependencies to: No dependencies.

FDP_ACF_EXT.3.1

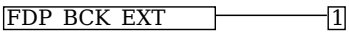
The TSF shall enforce an access control policy that prohibits an application from granting both write and execute permission to a file on the device except for [**selection**: *files stored in the application's private data folder*, *no exceptions*].

B.2.7.2 FDP_BCK_EXT Application Backup

Family Behavior

This family defines requirements for managing device backups.

Component Leveling



FDP_BCK_EXT.1, Application Backup, requires the TSF to be able to determine which data to include in backup operations.

Management: FDP_BCK_EXT.1

The following actions could be considered for the management functions in FMT:

- Enable/disable backup of certain applications to a local or remote system.

Audit: FDP_BCK_EXT.1

There are no auditable events foreseen.

FDP_BCK_EXT.1 Application Backup

Hierarchical to: No other components.

Dependencies to: No dependencies.

FDP_BCK_EXT.1.1

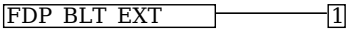
The TSF shall provide a mechanism for applications to mark [**assignment**: *list of data categories*] to be excluded from device backups.

B.2.7.3 FDP_BLT_EXT Limitation of Bluetooth Device Access

Family Behavior

This family defines requirements for managing Bluetooth devices.

Component Leveling



FDP_BLT_EXT.1, Limitation of Bluetooth Device Access, requires the TSF to manage which applications communicate with Bluetooth devices.

Management: FDP_BLT_EXT.1

There are no management activities foreseen.

Audit: FDP_BLT_EXT.1

There are no auditable events foreseen.

FDP_BLT_EXT.1 Limitation of Bluetooth Device Access

Hierarchical to: No other components.

Dependencies to: No dependencies.

FDP_BLT_EXT.1.1

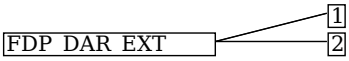
The TSF shall limit the applications that may communicate with a particular paired Bluetooth device.

B.2.7.4 FDP_DAR_EXT Data-at-Rest Encryption

Family Behavior

This family defines requirements for implementation of data-at-rest protection.

Component Leveling



FDP_DAR_EXT.1, Protected Data Encryption, requires the TSF to be able to protect all data with a chosen method of encryption.

FDP_DAR_EXT.2, Sensitive Data Encryption, requires the TSF to protect the Trust Anchor Database.

Management: FDP_DAR_EXT.1

The following actions could be considered for the management functions in FMT:

- Enabling data-at-rest protection.
- Enabling removable media's data-at-rest protection.

Audit: FDP_DAR_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Failure to encrypt/decrypt data.

FDP_DAR_EXT.1 Protected Data Encryption

Hierarchical to: No other components.

Dependencies to: FCS_COP.1 Cryptographic Operation

FDP_DAR_EXT.1.1

Encryption shall cover all protected data.

FDP_DAR_EXT.1.2

Encryption shall be performed using DEKs with AES in the [**assignment**: *list of AES modes*] mode with key size [**assignment**: *list of acceptable key sizes*] bits.

Management: FDP_DAR_EXT.2

The following actions could be considered for the management functions in FMT:

- Importing X.509v3 certificates into the Trust Anchor Database.
- Removing imported X.509v3 certificates from the Trust Anchor Database.
- Approving import and removal by applications of X.509v3 certificates in the Trust Anchor Database.

Audit: FDP_DAR_EXT.2

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Addition or removal of certificate from Trust Anchor Database

FDP_DAR_EXT.2 Sensitive Data Encryption

Hierarchical to: No other components.

Dependencies to: FCS_COP.1 Cryptographic Operation
FCS_CKM_EXT.7 Cryptographic Key Establishment
FCS_STG_EXT.2 Encrypted Cryptographic Key Storage

FDP_DAR_EXT.2.1

The TSF shall provide a mechanism for applications to mark data and keys as sensitive.

FDP_DAR_EXT.2.2

The TSF shall use an asymmetric key scheme to encrypt and store sensitive data received while the product is locked.

FDP_DAR_EXT.2.3

The TSF shall encrypt any stored symmetric key and any stored private key of the asymmetric keys used for the protection of sensitive data according to [**assignment**: *mechanism for encrypted key storage*].

FDP_DAR_EXT.2.4

The TSF shall decrypt the sensitive data that was received while in the locked state upon transitioning to the unlocked state using the asymmetric key scheme and shall re-encrypt that sensitive data using the symmetric key scheme.

B.2.7.5 FDP_IFC_EXT Subset Information Flow Control

Family Behavior

This family defines requirements for handling of information flows that are not addressed by FDP_IFC in CC Part 2.

Component Leveling

FDP_IFC_EXT.1, Subset Information Flow Control, requires the TSF to be able to support the use of an IPsec VPN to protect data in transit.

Management: FDP_IFC_EXT.1

The following actions could be considered for the management functions in FMT:

- Enabling/disabling VPN protection.
- Enabling/disabling Always On VPN protection.

Audit: FDP_IFC_EXT.1

There are no auditable events foreseen.

FDP_IFC_EXT.1 Subset Information Flow Control

Hierarchical to: No other components.

Dependencies to: FTP_ITC_EXT.1 Trusted Channel Communication

FDP_IFC_EXT.1.1

The TSF shall [**selection**:

- *provide an interface which allows a VPN client to protect all IP traffic using IPsec*
- *provide a VPN client which can protect all IP traffic using IPsec* ***as defined in the PP-Module for Virtual Private Network (VPN) Clients, version 3.0***

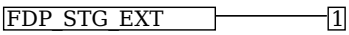
] with the exception of IP traffic needed to manage the VPN connection, and [**selection**: *[**assignment**: traffic needed for correct functioning of the TOE] , no other traffic*], when the VPN is enabled.

B.2.7.6 FDP_STG_EXT User Data Storage

Family Behavior

This family defines requirements for managing data storage.

Component Leveling



FDP_STG_EXT.1, User Data Storage, requires the TSF to be able to label, encrypt, store, and decrypt sensitive data and keys.

Management: FDP_STG_EXT.1

There are no management activities foreseen.

Audit: FDP_STG_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Failure to encrypt/decrypt data.

FDP_STG_EXT.1 User Data Storage

Hierarchical to: No other components.

Dependencies to: FCS_COP.1 Cryptographic Operation
FCS_STG_EXT.2 Encrypted Cryptographic Key Storage

FDP_STG_EXT.1.1

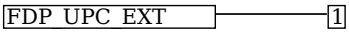
The TSF shall provide protected storage for the Trust Anchor Database.

B.2.7.7 FDP_UPC_EXT Inter-TSF User Data Transfer Protection

Family Behavior

This family defines requirements for the use of trusted channel protocols to protect user data.

Component Leveling



FDP_UPC_EXT.1, Inter-TSF User Data Transfer Protection, requires the TSF to be able to protect communication channels between products using a chosen secure method.

Management: FDP_UPC_EXT.1

There are no management activities foreseen.

Audit: FDP_UPC_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Application initiation of trusted channel.

FDP_UPC_EXT.1 Inter-TSF User Data Transfer Protection

Hierarchical to: No other components.

Dependencies to: FTP_ITC_EXT.1 Trusted Channel Communication

FDP_UPC_EXT.1.1

The TSF shall provide a means for non-TSF applications executing on the TOE to use [**assignment**: *data transfer protocol*] to provide a protected communication channel between the non-TSF application and another IT product that is logically distinct from other communication channels, provides assured identification of its end points, protects channel data from disclosure, and detects modification of the channel data.

FDP_UPC_EXT.1.2

The TSF shall permit the non-TSF applications to initiate communication via the trusted channel.

Appendix C - Implicitly Satisfied Requirements

This appendix lists requirements that should be considered satisfied by products successfully evaluated against this PP. These requirements are not featured explicitly as SFRs and should not be included in the ST. They are not included as standalone SFRs because it would increase the time, cost, and complexity of evaluation. This approach is permitted by [CC] Part 1, 8.3 Dependencies between components.

This information benefits systems engineering activities which call for inclusion of particular security controls. Evaluation against the PP provides evidence that these controls are present and have been evaluated.

Requirement	Rationale for Satisfaction
FAU_SEL.1 - Selective Audit	FAU_SEL.1 has a dependency on FMT_MTD.1 since configuration of audit data is a subset of managing TSF data. This dependency is met by FMT_SMF_EXT.1, which defines "configure the auditable items" as a management function and specifies the roles that may perform this, consistent with how FMT_MTD.1 would typically satisfy the dependency.
FCS_STG_EXT.1 - Cryptographic Key Storage	FCS_STG_EXT.1 has a dependency on FMT_SMR.1 for the management roles that are authorized to manage the functionality defined by the requirement. This dependency is met by FMT_SMF_EXT.1, which implicitly defines separate management roles for the TSF.
FDP_ACF_EXT.1 - Access Control for System Services	FDP_ACF_EXT.1 has a dependency on FMT_SMR.1 for the management roles that are authorized to manage the functionality defined by the requirement. This dependency is met by FMT_SMF_EXT.1, which implicitly defines separate management roles for the TSF.
FDP_ACF_EXT.2 - Access Control for System Resources	FDP_ACF_EXT.2 has a dependency on FMT_SMR.1 for the management roles that are authorized to manage the functionality defined by the requirement. This dependency is met by FMT_SMF_EXT.1, which implicitly defines separate management roles for the TSF.
FIA_AFL_EXT.1 - Authentication Failure Handling	FIA_AFL_EXT.1 has a dependency on FIA_UAU.1 since handling of authentication failures is not possible without an authentication mechanism. This dependency is met by the extended SFR FIA_UAU_EXT.1, which serves the same purpose as its CC Part 2 equivalent.
FIA_PMG_EXT.1 - Password Management	FIA_PMG_EXT.1 has a dependency on FIA_UAU.1 since composition of authentication credentials is not possible without an authentication mechanism. This dependency is met by the extended SFR FIA_UAU_EXT.1, which serves the same purpose as its CC Part 2 equivalent.
FIA_UAU.7 - Protected Authentication Feedback	FIA_UAU.7 has a dependency on FIA_UAU.1 since protected authentication feedback is not possible without an authentication mechanism. This dependency is met by the extended SFR FIA_UAU_EXT.1, which serves the same purpose as its CC Part 2 equivalent.
FMT_MOF_EXT.1 - Management of Security Functions Behavior	FMT_MOF_EXT.1 has a dependency on FMT_SMF.1 through its reference to management functions in the requirement text. This dependency is met by FMT_SMF_EXT.1, which defines management functions specifically for the TSF.
FMT_SMF_EXT.3 - Current Administrator	FMT_SMF_EXT.3 has a dependency on FMT_SMR.1 through its reference to management roles in the requirement text. This dependency is met by FMT_SMF_EXT.1, which implicitly defines separate management roles for the TSF.

Appendix D - Entropy Documentation And Assessment

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

D.1 Design Description

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

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

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

D.2 Entropy Justification

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

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

D.3 Operating Conditions

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

D.4 Health Testing

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

Appendix E - Acronyms

Table 24: Acronyms

Acronym	Meaning
AEAD	Authenticated Encryption with Associated Data
AES	Advanced Encryption Standard
ANSI	American National Standards Institute
AP	Application Processor
API	Application Programming Interface
ASLR	Address Space Layout Randomization
BAF	Biometric Authentication Factor
Base-PP	Base Protection Profile
BP	Baseband Processor
BR/EDR	(Bluetooth) Basic Rate/Enhanced Data Rate
BYOD	Bring Your Own Device
CA	Certificate Authority
CBC	Cipher Block Chaining
CC	Common Criteria
CCM	Counter with CBC-Message Authentication Code
CCMP	CCM Protocol
CEM	Common Evaluation Methodology
CMC	Certificate Management over Cryptographic Message Syntax (CMS)
cPP	Collaborative Protection Profile
CPU	Central Processing Unit
CRL	Certificate Revocation List
CSP	Critical Security Parameter
DAR	Data At Rest
DEK	Data Encryption Key
DEK	Data Encryption Key
DEP	Data Execution Prevention
DH	Diffie-Hellman
DNS	Domain Name System
DSA	Digital Signature Algorithm
DTLS	Datagram Transport Layer Security
EAP	Extensible Authentication Protocol
EAPOL	EAP Over LAN
ECDH	Elliptic Curve Diffie Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
EEPROM	Electrically Erasable Programmable Read-Only Memory
EP	Extended Package
EST	Enrollment over Secure Transport
FEK	File Encryption Key
FFC	Finite Field Cryptography
FIPS	Federal Information Processing Standards

FM	Frequency Modulation
FP	Functional Package
FQDN	Fully Qualified Domain Name
GCM	Galois Counter Mode
GPS	Global Positioning System
GPU	Graphics Processing Unit
HDMI	High Definition Multimedia Interface
HMAC	Keyed-Hash Message Authentication Code
HTTPS	HyperText Transfer Protocol Secure
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
IPC	Inter-Process Communication
IPsec	Internet Protocol Security
KAT	Known Answer Test
KDF	Key Derivation Function
KEK	Key Encryption Key
LE	(Bluetooth) Low Energy
LTE	Long Term Evolution
MD	Mobile Device
MDM	Mobile Device Management
MMI	Man-Machine Interface
MMS	Multimedia Messaging Service
MMU	Memory Management Unit
NFC	Near Field Communication
NIST	National Institute of Standards and Technology
NTP	Network Time Protocol
NX	Never Execute
OCSP	Online Certificate Status Protocol
OE	Operational Environment
OID	Object Identifier
OS	Operating System
OTA	Over the Air
PAE	Port Access Entity
PBKDF	Password-Based Key Derivation Function
PD	Protected Data
PIV	Personal Identity Verification
PMK	Pairwise Master Key
PP	Protection Profile
PP-Configuration	Protection Profile Configuration
PP-Module	Protection Profile Module
PRF	Pseudorandom Function
PSK	Pre-Shared Key
PTK	Pairwise Temporal Key
RA	Registration Authority

RBG	Random Bit Generator
REK	Root Encryption Key
ROM	Read-only memory
RSA	Rivest Shamir Adleman Algorithm
SAFAR	System Authentication False Accept Rate
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SHA	Secure Hash Algorithm
SMS	Short Messaging Service
SoC	System On a Chip
SPI	Security Parameter Index
SSH	Secure Shell
SSID	Service Set Identifier
ST	Security Target
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Functionality
TSFI	TSF Interface
TSS	TOE Summary Specification
URI	Uniform Resource Identifier
USB	Universal Serial Bus
USSD	Unstructured Supplementary Service Data
VPN	Virtual Private Network
XCCDF	eXtensible Configuration Checklist Description Format
XTS	XEX (XOR Encrypt XOR) Tweakable Block Cipher with Ciphertext Stealing

Appendix F - Bibliography

Table 25: Bibliography

Identifier	Title
[CC]	Common Criteria for Information Technology Security Evaluation - • Part 1: Introduction and general model, CCMB-2022-11-001, CC:2022, Revision 1, November 2022. • Part 2: Security functional requirements, CCMB-2022-11-002, CC:2022, Revision 1, November 2022. • Part 3: Security assurance requirements, CCMB-2022-11-003, CC:2022, Revision 1, November 2022. • Part 4: Framework for the specification of evaluation methods and activities, CCMB-2022-11-004, CC:2022, Revision 1, November 2022. • Part 5: Pre-defined packages of security requirements, CCMB-2022-11-005, CC:2022, Revision 1, November 2022.
[CEM]	Common Methodology for Information Technology Security Evaluation - • Evaluation methodology, CCMB-2022-11-006, CC:2022, Revision 1, November 2022.

Appendix G - Acknowledgments

This protection profile was developed by the Mobility Technical Community with representatives from industry, U.S. Government agencies, Common Criteria Test Laboratories, and international Common Criteria schemes. The National Information Assurance Partnership wishes to acknowledge and thank the members of this group whose dedicated efforts contributed significantly to the publication. These organizations include:

U.S. Government
Defense Information Systems Agency (DISA)
CyberSecurity Directorate (CSD)
National Information Assurance Partnership (NIAP)
National Institute of Standards and Technology (NIST)

International Common Criteria Schemes
Australian Information Security Evaluation Program (AISEP)
Canadian Common Criteria Evaluation and Certification Scheme (CSEC)
Information-technology Promotion Agency, Japan (IPA)
UK IT Security Evaluation and Certificate Scheme (NCSC)

Industry
Apple, Inc.
BlackBerry
LG Electronics, Inc.
Microsoft Corporation
Motorola Solutions
Samsung Electronics Co., Ltd.
Other Members of the Mobility Technical Community

Common Criteria Test Laboratories
EWA-Canada, Ltd.
Gossamer Security Solutions