Protection Profile for Mobile Device Fundamentals

Version 1.0 21 October 2013

Acknowledgements

This protection profile was developed by the Mobility Technical Community with representatives from industry, U.S. Government agencies, 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)

Information Assurance Directorate (IAD)

National Information Assurance Partnership (NIAP)

National Institute of Standards and Technology (NIST)

International Common Criteria Schemes

Australasian Information Security Evaluation Program (AISEP)

Canadian Common Criteria Evaluation and Certification Scheme (CSEC)

UK IT Security Evaluation and Certificate Scheme (CESG)

Industry

Apple, Inc.

BlackBerry

Microsoft Corporation

Motorola Solutions

Samsung Electronics Co., Ltd.

Other Members of the Mobility Technical Community

0. Preface

0.1 Objectives of Document

This document presents the Common Criteria (CC) Protection Profile (PP) to express the fundamental security and evaluation requirements for a mobile device.

0.2 Scope of Document

The scope of the Protection Profile within the development and evaluation process is described in the Common Criteria for Information Technology Security Evaluation [CC]. In particular, a PP defines the IT security requirements of a generic type of TOE and specifies the functional and assurance security measures to be offered by that TOE to meet stated requirements [CC1, Section C.1].

0.3 Intended Readership

The target audiences of this PP are mobile device developers, CC consumers, evaluators and schemes.

0.4 Related Documents

Common Criteria[1]

[CC1] Common Criteria for Information Technology Security Evaluation,
Part 1: Introduction and General Model,
CCMB-2012-09-001, Version 3.1 Revision 4, September 2012.

[CC2] Common Criteria for Information Technology Security Evaluation,
Part 2: Security Functional Components,
CCMB-2012-09-002, Version 3.1 Revision 4, September 2012.

[CC3] Common Criteria for Information Technology Security Evaluation,
Part 3: Security Assurance Components,
CCMB-2012-09-003, Version 3.1 Revision 4, September 2012.

[CEM] Common Methodology for Information Technology Security Evaluation,
Evaluation Methodology,
CCMB-2012-09-004, Version 3.1, Revision 4, September 2012.

[1] For details see http://www.commoncriteriaportal.org/

Contents

Figures / Tables

1. PP Introduction

1.1 PP Reference Identification

PP Reference: Protection Profile for Mobile Device Fundamentals

PP Version: 1.0

PP Date: 21 October 2013

1.2 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.

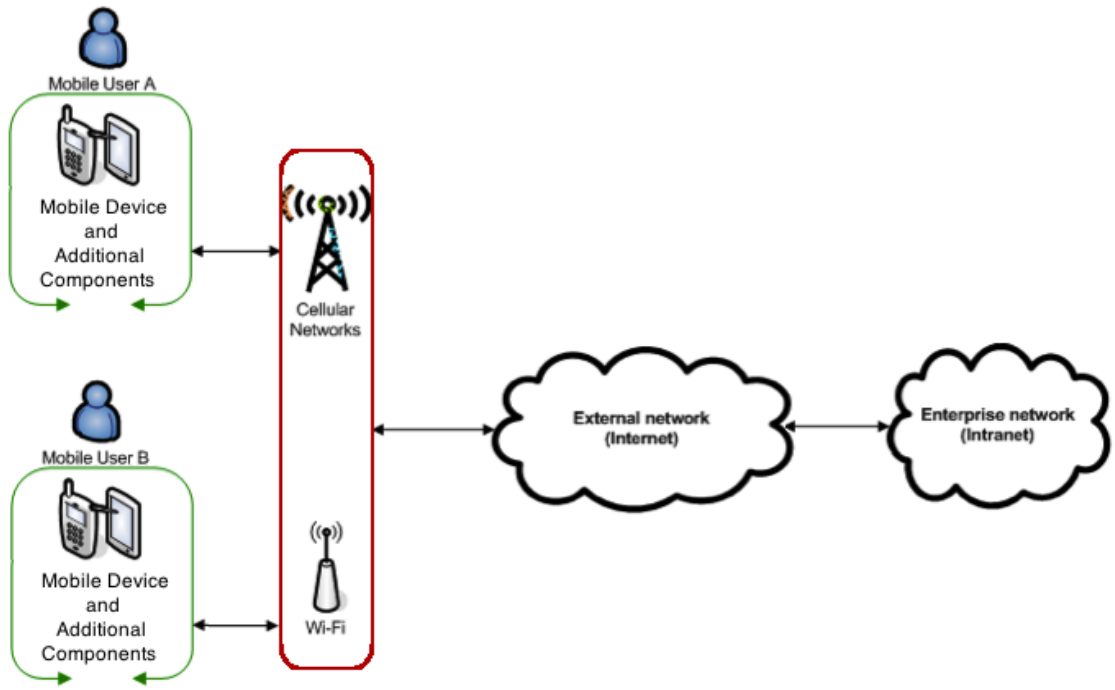

Figure 1: Mobile Device Network Environment

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. As illustrated in Figure 2, it is expected that a typical deployment would also include either third-party or bundled components that provide:

- Data in transit protection (e.g. VPN Client, VoIP Client, Web Browser)
- Security policy management (e.g. MDM System)

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. 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 VoIP client, email client, and web browser.

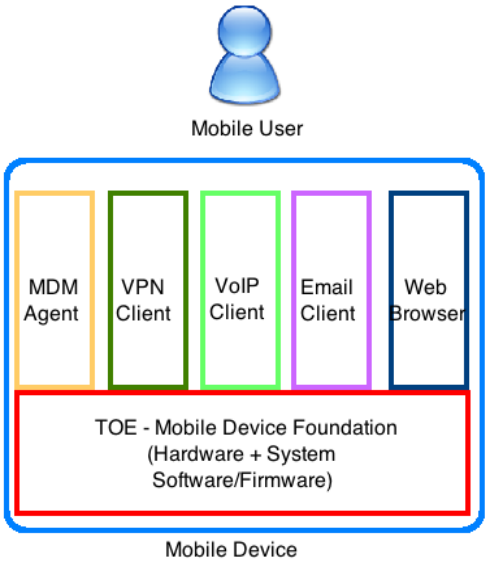

Figure 2: Optional Additional Mobile Device Components

1.3 TOE Usage

The mobile device may be operated in a number of use cases. 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.

Annex G provides use case templates that list those selections, assignments, and objective requirements that best support the use cases identified by this Protection Profile. Several of the use cases templates include objective requirements that are strongly desired for the indicated use cases. Readers can expect those requirements to be made mandatory in the next

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 the main body is sufficient 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 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 WiFi 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 would not be considered realistic 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 which is used for both personal activities and enterprise data is commonly called Bring Your Own Device (BYOD). 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 certain security policies, for example a password or screenlock policy, and may require assured enterprise software, for example a VPN client, before allowing access. The device may be provisioned for access to enterprise resources after significant personal usage has occurred.

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

A personally-owned device may also be given access to limited enterprise services 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.

2. CC Conformance

As defined by the references [CC1], [CC2] and [CC3], this PP conforms to the requirements of Common Criteria v3.1, Revision 4. The methodology applied for the PP evaluation is defined in [CEM].

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 Protection Profile are those of network eavesdropping, network attacks, physical access, and malicious or flawed applications, as detailed in the following sections.

3.1.1 T.EAVESDROP Network Eavesdropping

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.

3.1.2 T.NETWORK 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.

3.1.3 T.PHYSICAL Physical Access

The loss or theft of the Mobile Device may give rise to loss of confidentiality of user data including credentials. These physical access threats may involve attacks which attempt to access the device through external hardware ports, through its user interface, and also through direct and possibly destructive access to its storage media. The goal of such attacks is to access data from a lost or stolen device which is not expected to return to its user.

Note: Defending against device re-use after physical compromise is out of scope for this protection profile.

3.1.4 T.FLAWAPP Malicious or Flawed Application

Applications loaded onto the Mobile Device may include malicious or exploitable code. This code could be included intentionally by its developer 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.

3.1.5 T.PERSISTENT Persistent Access

Persistent access to 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 at least as well as by its legitimate owner.

3.2 Assumptions

The assumptions for the Mobile Device are defined in Annex A.1.1.

3.3 Organizational Security Policy

There are no OSPs for the Mobile Device.

4. Security Objectives

4.1 Security Objectives for the TOE

The security objectives for the Mobile Device are defined as follows.

4.1.1 O.COMMS Protected Communications

To address the network eavesdropping and network attack threats described in Section 3.1, concerning wireless transmission of Enterprise and user data and configuration data between the TOE and remote network entities, conformant TOEs will use a trusted communication path. The TOE will be capable of communicating using one (or more) of these standard protocols: IPsec, DTLS, TLS, or HTTPS. The protocols are specified by RFCs that offer a variety of implementation choices. Requirements have been imposed on some of these choices (particularly those for cryptographic primitives) to provide interoperability and resistance to cryptographic attack.

While conformant TOEs must support all of the choices specified in the ST, they may support additional algorithms and protocols. If such additional mechanisms are not evaluated, guidance must be given to the administrator to make clear the fact that they were not evaluated.

> FCS_CKM.1(*), FCS_CKM.2, FCS_COP.1(*), FCS_DTLS_EXT.1, FCS_HTTPS_EXT.1, FCS_RBG_EXT.1, FCS_SRV_EXT.1, FCS_TLS_EXT.1, FCS_TLS_EXT.2, FDP_STG_EXT.1, FIA_PAE_EXT.1, FIA_X509_EXT.1, FIA_X509_EXT.2, FIA_X509_EXT.3, FTA_WSE_EXT.1, FTP_ITC_EXT.1

4.1.2 O.STORAGE Protected Storage

To address the issue of loss of confidentiality of user data in the event of loss of a mobile device (T.PHYSICAL), conformant TOEs will use data-at-rest protection. The TOE will be capable of encrypting data and keys stored on the device and will prevent unauthorized access to encrypted data.

> FCS_CKM_EXT.1, FCS_CKM_EXT.2, FCS_CKM_EXT.3, FCS_CKM_EXT.4, FCS_CKM_EXT.5, FCS_CKM_EXT.6, FCS_COP.1(*), FCS_IV_EXT.1, FCS_RBG_EXT.1, FCS_STG_EXT.1, FCS_STG_EXT.2, FCS_STG_EXT.3, FDP_DAR_EXT.1, FDP_DAR_EXT.2, FIA_UAU_EXT.1, FPT_KST_EXT.1, FPT_KST_EXT.2, FPT_KST_EXT.3

4.1.3 O.CONFIG Mobile Device Configuration

To ensure a mobile device protects user and enterprise data that it may store or process, conformant TOEs will provide the capability to configure and apply security policies defined by the user and the Enterprise Administrator. If Enterprise security policies are configured these must be applied in precedence of user specified security policies.

> FMT_MOF.1(*), FMT_POL_EXT.1, FMT_SMF.1, FMT_SMF_EXT.1, FTA_TAB_EXT.1

4.1.4 O.AUTH Authorization and Authentication

To address the issue of loss of confidentiality of user data in the event of loss of a mobile device (T.PHYSICAL), users are required to enter an authentication factor to the device prior to accessing protected functionality and data. Some non-sensitive functionality (e.g., emergency calling, text notification) can be accessed prior to entering the authentication factor. The device will automatically lock following a configured period of inactivity in an attempt to ensure authorization will be required in the event of the device being lost or stolen.

Authentication of the endpoints of a trusted communication path is required for network access to ensure attacks are unable to establish unauthorized network connections to undermine the integrity of the device.

Repeated attempts by a user to authorize to the TSF will be limited or throttled to enforce a delay between unsuccessful attempts.

FCS_CKM.1(2), FIA_AFL_EXT.1, FIA_BLT_EXT.1, FIA_PMG_EXT.1, FIA_TRT_EXT.1, FIA_UAU.7, FIA_UAU_EXT.1, FIA_UAU_EXT.2, FIA_UAU_EXT.3, FIA_X509_EXT.2, FTA_SSL_EXT.1

4.1.5 O.INTEGRITY Mobile Device Integrity

To ensure the integrity of the mobile device is maintained conformant TOEs will perform self-tests to ensure the integrity of critical functionality, software/firmware and data has been maintained. The user shall be notified of any failure of these self-tests. (This will protect against the threat T.PERSISTENT.)

To address the issue of an application containing malicious or flawed code (T.FLAWAPP), the integrity of downloaded updates to software/firmware will be verified prior to installation/execution of the object on the TOE. In addition, the operating system will restrict applications to only have access to the system services and data they are permitted to interact with. The operating system will further protect against malicious applications from gaining access to data they are not authorized to access by randomizing the memory layout.

FAU_GEN.1, FAU_SEL.1, FAU_STG_EXT.1, FCS_COP.1(2), FCS_COP.1(3), FCS_ACF_EXT.1, FPT_AEX_EXT.1, FPT_AEX_EXT.2, FPT_AEX_EXT.3, FPT_AEX_EXT.4, FPT_BBD_EXT.1, FPT_NOT_EXT.1, FPT_STM.1, FPT_TST_EXT.1, FPT_TST_EXT.2, FPT_TUD_EXT.1, FPT_TUD_EXT.2

4.2 Security Objectives for the Operational Environment

The objectives that are required to be met by the TOE’s operational environment are defined in Annex A.2.2.

5. Security Functional Requirements

The individual security functional requirements are specified in the sections below.

5.1 Conventions

The following conventions are used for the completion of operations:

- [*Italicized text within square brackets*] indicates an operation to be completed by the ST author
- ~~Strikethrough~~ indicates text removed as a refinement and underlined text indicates additional text provided as a refinement.
- [**Bold text within square brackets**] indicates the completion of an assignment.
- [***Bold-italicized text within square brackets***] indicates the completion of a selection.

5.2 Class: Cryptographic Support (FCS)

5.2.1 Cryptographic Key Management (FCS_CKM)

This section describes how keys are generated, derived, combined, 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 Section 5.3.2). 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.

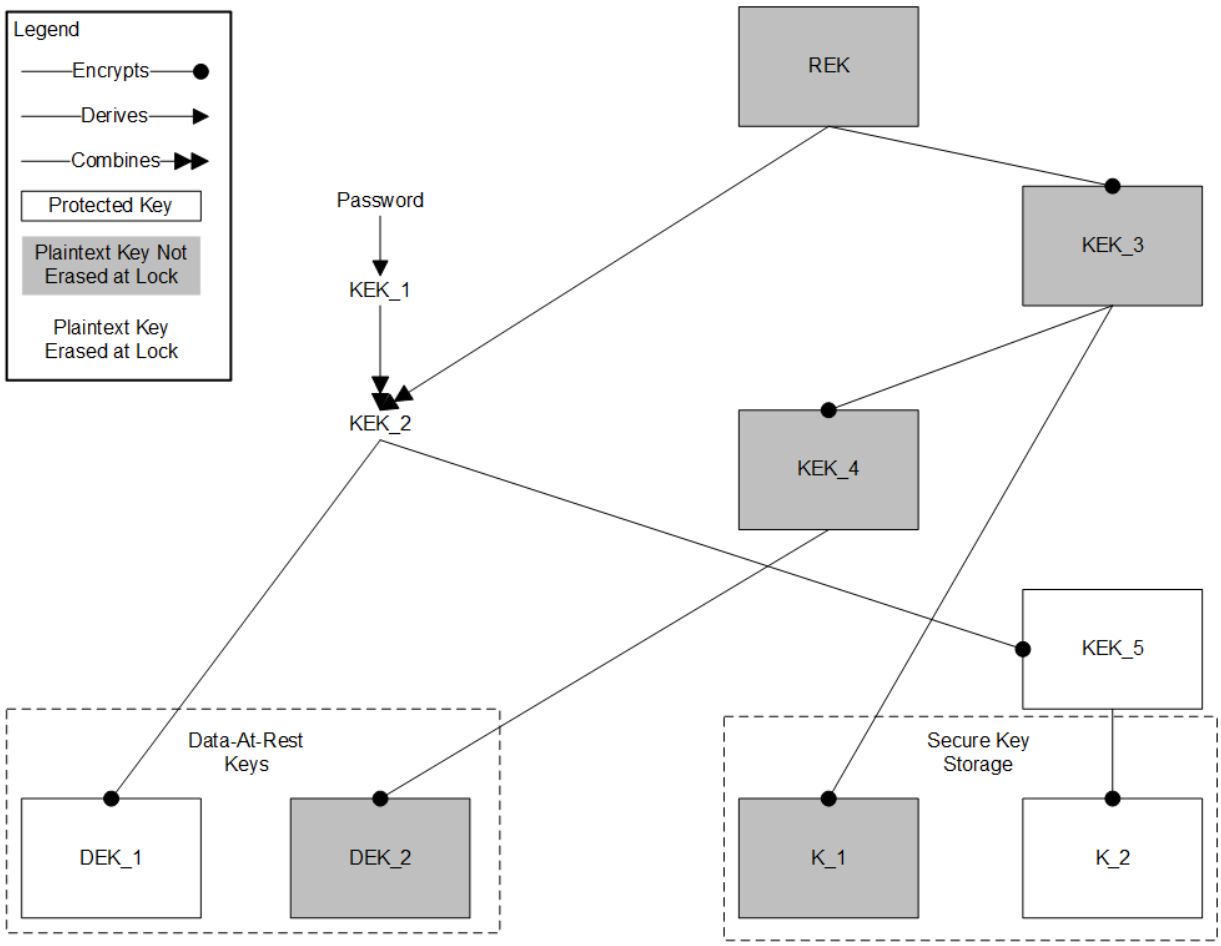

Figure 3: An Illustrative Key Hierarchy

5.2.1.1 Cryptographic Key Generation (Key Establishment)

FCS_CKM.1(1)	Refinement: Cryptographic key generation

FCS_CKM.1.1(1): The TSF shall generate asymmetric cryptographic keys used for key establishment in accordance with:

- NIST Special Publication 800-56B, "Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography" for RSA-based key establishment schemes and

[selection:

- *NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" for finite field-based key establishment schemes;*
- *NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" for elliptic curve-based key establishment schemes and implementing "NIST curves" P-256, P-384 and*

[selection: P-521, no other curves] (as defined in FIPS PUB 186-4, "Digital Signature Standard")

- *No other algorithms*]

and specified cryptographic key sizes equivalent to, or greater than, a symmetric key strength of 112 bits.

Application Note: This component requires that the TOE be able to generate the public/private key pairs that are used for key establishment purposes for the various cryptographic protocols used by the TOE (e.g., EAP-TLS, trusted channel) and for data-at-rest protection at screenlock. If multiple schemes are supported, then the ST author should iterate this requirement to capture this capability. The scheme used will be chosen by the ST author from the selection. RSA Key establishment is required in accordance with **FCS_TLS_EXT.1**.

The domain parameters to be used may be specified by the requirements of the protocol in this PP. In general, it is not expected that the TOE will generate domain parameters, and therefore there is no additional domain parameter validation needed when the TOE complies with the protocols and other requirements specified in this PP.

The generated key strength of 2048-bit DSA and RSA keys need to be equivalent to, or greater than, a symmetric key strength of 112 bits. See NIST Special Publication 800-57, "Recommendation for Key Management" for information about equivalent key strengths.

In the future, NIST SP 800-56A for elliptic curves will be required.

Assurance Activity:
This assurance activity will verify the key generation and key establishments schemes used on the TOE.
Key Generation:
The evaluator shall verify the implementation of the key generation routines of the supported schemes using the applicable tests below.
Key Generation for RSA-Based Key Establishment Schemes

The evaluator shall verify the implementation of RSA Key Generation by the TOE using the Key Generation test. This test verifies the ability of the TSF to correctly produce values for the key components including the public verification exponent *e*, the private prime factors *p* and *q*, the public modulus *n* and the calculation of the private signature exponent *d*.

Key Pair generation specifies 5 ways (or methods) to generate the primes *p* and *q*. These include:

1. Random Primes:
   - Provable primes
   - Probable primes
2. Primes with Conditions:
   - Primes *p1, p2, q1,q2, p* and *q* shall all be provable primes
   - Primes *p1, p2, q1,* and *q2* shall be provable primes and *p* and *q* shall be probable primes
   - Primes *p1, p2, q1,q2, p* and *q* shall all be probable primes

To test the key generation method for the Random Provable primes method and for all the Primes with Conditions methods, the evaluator must seed the TSF key generation routine with sufficient data to deterministically generate the RSA key pair. This includes the random seed(s), the public exponent of the RSA key, and the desired key length. For each key length supported, 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 from a known good implementation.

Key Generation for Finite-Field Cryptography (FFC) – Based 56A Schemes

FFC Domain Parameter and Key Generation Tests

The evaluator shall verify the implementation of the Parameters Generation and the Key Generation for FFC by the TOE using the Parameter Generation and Key Generation test. This test verifies the ability of the TSF to correctly produce values for the field prime p, the cryptographic prime q (dividing p-1), the cryptographic group generator g, and the calculation of the private key x and public key y.

The Parameter generation specifies 2 ways (or methods) to generate the cryptographic prime q and the field prime p:

Cryptographic and Field Primes:

- Primes q and p shall both be provable primes
- Primes q and field prime p shall both be probable primes

and two ways to generate the cryptographic group generator g:

Cryptographic Group Generator:

- Generator g constructed through a verifiable process
- Generator g constructed through an unverifiable process.

The Key generation specifies 2 ways to generate the private key x:

Private Key:

- len(q) bit output of RBG where 1 <=x <= q-1
- len(q) + 64 bit output of RBG, followed by a mod q-1 operation where 1<= x<=q-1.

The security strength of the RBG must be at least that of the security offered by the FFC parameter set.

To test the cryptographic and field prime generation method for the provable primes method and/or the group generator g for a verifiable process, the evaluator must seed the TSF parameter generation routine with sufficient data to deterministically generate the parameter set.

For each key length supported, the evaluator shall have the TSF generate 25 parameter sets and key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation. Verification must also confirm

- g != 0,1
- q divides p-1
- g^q mod p = 1
- g^x mod p = y

for each FFC parameter set and key pair.

Key Generation for Elliptic Curve Cryptography (ECC) - Based 56A Schemes

ECC Key Generation Test

For each supported NIST curve, i.e., P-256, P-284 and P-521, the evaluator shall require the implementation under test (IUT) to generate 10 private/public key pairs. The private key shall be generated using an approved random bit generator (RBG). To determine correctness, the evaluator shall submit the generated key pairs to the public key verification (PKV) function of a known good implementation.

ECC Public Key Verification (PKV) Test

For each supported NIST curve, i.e., P-256, P-284 and P-521, the evaluator shall generate 10 private/public key pairs using the key generation function of a known good implementation and modify five of the public key values so that they are incorrect, leaving five values unchanged (i.e., correct). The evaluator shall obtain in response a set of 10 PASS/FAIL values.

Key Establishment Schemes

The evaluator shall verify the implementation of the key establishment schemes of the supported by the TOE using the applicable tests below.

SP800-56A Key Establishment Schemes

The evaluator shall verify a TOE's implementation of SP800-56A key agreement schemes 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 specifications in the Recommendation. These components include the calculation of the DLC primitives (the shared secret value Z) and the calculation of the derived keying material (DKM) via the Key Derivation Function (KDF). If key confirmation is supported, the evaluator shall also verify that the components of key confirmation have been implemented correctly, using the test procedures described below. This includes the parsing of the DKM, the generation of MACdata and the calculation of MACtag.

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 scheme-key agreement role combination, KDF type, and, if supported, key confirmation role- key confirmation type combination, the tester shall generate 10 sets of test vectors. The data set consists of one set of domain parameter values (FFC) or the NIST approved curve (ECC) per 10 sets of public keys. These keys are static, ephemeral or both depending on the scheme being tested.

The evaluator shall obtain the DKM, the corresponding TOE's public keys (static and/or ephemeral), the MAC tag(s), and any inputs used in the KDF, such as the Other Information field OI and TOE id fields.

If the TOE does not use a KDF defined in SP 800-56A, the evaluator shall obtain only the public keys and the hashed value of the shared secret.

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, derive the keying material DKM, and compare hashes or MAC tags generated from these values.

If key confirmation is supported, the TSF shall perform the above for each implemented approved MAC algorithm.

Validity Test

The Validity test verifies the ability of the TOE to recognize another party's valid and invalid key agreement results with or without key confirmation. To conduct this test, the evaluator shall obtain a list of the supporting cryptographic functions included in the SP800-56A key agreement implementation to determine which errors the TOE should be able to recognize. The evaluator generates a set of 24 (FFC) or 30 (ECC) test vectors consisting of data sets including domain parameter values or NIST approved curves, the evaluator's public keys, the TOE's public/private key pairs, MACTag, and any inputs used in the KDF, such as the other info and TOE id fields.

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 Z, the DKM, the other information field OI, the data to be MACed, or the generated MACTag. If the TOE contains the full or partial (only ECC) public key validation, the evaluator will also individually inject errors in both parties' static public keys, both parties' ephemeral public keys and the TOE's static private key to assure the TOE detects errors in the public key validation function and/or the partial key validation function (in ECC only). 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.

SP800-56B Key Establishment Schemes

At this time, detailed test procedures for RSA-based key establishment schemes are not available. In order to show that the TSF complies with 800-56A and/or 800-56B, depending on the selections made, the evaluator shall ensure that the TSS contains the following information:

- The TSS shall list all sections of the appropriate 800-56 standard(s) to which the TOE complies.
- For each applicable section listed in the TSS, for all statements that are not "shall" (that is, "shall not", "should", and "should not"), if the TOE implements such options it shall be described in the TSS. If the included functionality is indicated as "shall not" or "should not" in the standard, the TSS shall provide a rationale for why this will not adversely affect the security policy implemented by the TOE.

For each applicable section of 800-56A and 800-56B (as selected), any omission of functionality related to "shall" or "should" statements shall be described.

5.2.1.2 Cryptographic Key Generation (Asymmetric Keys for Authentication)

FCS_CKM.1(2)	Cryptographic key generation

FCS_CKM.1.1(2) The TSF shall generate asymmetric cryptographic keys used for authentication in accordance with a specified cryptographic key generation algorithm [selection:

- *FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3 for RSA schemes;*
- *FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4 for ECDSA schemes and implementing "NIST curves" P-256, P-384 and [selection: P-521, no other curves];*
- *ANSI X9.31-1998, Appendix A.2.4 Using AES for RSA schemes]*

and specified cryptographic key sizes [***equivalent to, or greater than, a symmetric key strength of 112 bits***].

Application Note: While it is expected that the public key generated be associated with an identity in an X509v3 certificate, this association is not required to be performed by the TOE, and instead is expected to be performed by a Certificate Authority in the Operational Environment.

For elliptic curve-based schemes, the key size refers to the $\log_2$ of the order of the base point.

The ANSI X9.31-1998 option will be removed from the selection in a future publication of this document. Presently, the selection is not exclusively limited to the FIPS PUB 186-4 options in order to allow industry some further time to complete the transition to the modern FIPS PUB 186-4 standard. As the preferred approach for cryptographic signature, elliptic curves will be required in future publications of this PP.

See NIST Special Publication 800-57, "Recommendation for Key Management" for information about equivalent key strengths.

Assurance Activity:
If the TSF implements a FIPS 186-4 signature scheme, this requirement is verified under **FCS_COP.1.1(3).**

If the ESF implements the ANSI X9.31-1998 scheme, the evaluator shall check to ensure that the TSS describes how the key-pairs are generated. In order to show that the TSF implementation complies with ANSI X9.31-1998, the evaluator shall ensure that the TSS contains the following information:

- The TSS shall list all sections of the standard to which the TOE complies;
- For each applicable section listed in the TSS, for all statements that are not "shall" (that is, "shall not", "should", and "should not"), if the TOE implements such options it shall be described in the TSS. If the included functionality is indicated as "shall not" or "should not" in the standard, the TSS shall provide a rationale for why this will not adversely affect the security policy implemented by the TOE;

- For each applicable section of Appendix B, any omission of functionality related to "shall" or “should” statements shall be described.

5.2.1.3 Cryptographic Key Generation (WLAN)

FCS_CKM.1(3)	Cryptographic key generation

FCS_CKM.1.1(3) The TSF shall generate symmetric cryptographic keys in accordance with a specified cryptographic key generation algorithm [***PRF-384***] and specified cryptographic key sizes [***128 bits***] using a Random Bit Generator as specified in **FCS_RBG_EXT.1** that meet the following: [***IEEE 802.11-2012***].

Application Note: This requirement supports **FTA_WSE_EXT.1**, which requires that the mobile device be able to connect to configured Wireless LANs. The cryptographic key derivation algorithm required by IEEE 802.11-2012 (Section 11.6.1.2) and verified in WPA2 certification is PRF-384 which uses the HMAC-SHA-1 function and outputs 384 bits.

This requirement applies only to the keys that are generated/derived for the communications between the access point and the client once the client has been authenticated. It refers to the derivation of the PTK from the PMK, which is done using a random value generated by the RBG specified in this PP, the HMAC function using SHA-1 as specified in this PP, as well as other information. This is specified in IEEE 802.11-2012 primarily in chapter 11.

Assurance Activity:

The cryptographic primitives will be verified through assurance activities specified later in this PP. The evaluator shall verify that the TSS describes how the primitives defined and implemented by this PP are used by the TOE in establishing and maintaining secure connectivity to the wireless clients. The TSS shall also provide a description of the developer’s method(s) of assuring that their implementation conforms to the cryptographic standards; this includes not only testing done by the developing organization, but also any third-party testing that is performed (e.g. WPA2 certification). The evaluator shall ensure that the description of the testing methodology is of sufficient detail to determine the extent to which the details of the protocol specifics are tested.

5.2.1.4 Cryptographic Key Distribution (WLAN)

FCS_CKM.2	Cryptographic key distribution

FCS_CKM.2.1 The TSF shall decrypt Group Temporal Key (GTK) in accordance with a specified cryptographic key distribution method [***AES Key Wrap in an EAPOL-Key frame***] that meets the following: [***NIST SP 800-38F, IEEE 802.11-2012 for the packet format and timing considerations***] and does not expose the cryptographic keys.

Application Note: This requirement supports **FTA_WSE_EXT.1**, which requires that the mobile device be able to connect to configured Wireless LANs. This requirement applies to the GTK that is received by the TOE for use in decrypting broadcast and multicast messages from the Access Point to which it's connected. IEEE 802.11-2012 specifies the format for the transfer as well as the fact that it must be wrapped by the AES Key Wrap method specified in NIST SP 800-38F; the TOE must be capable of unwrapping such keys.

Assurance Activity:

The evaluator shall check the TSS to ensure that it describes how the GTK is unwrapped prior to being installed for use on the TOE using the AES implementation specified in this PP. The evaluator shall also perform the following tests:

Test 1: The evaluator shall successfully connect the TOE to the access point. As the TOE is connected, the evaluator shall observe that the GTK is not transmitted in the clear between the TOE and the Access Point.

Test 2: The evaluator shall cause a broadcast message to be sent by the Access Point to which the TOE is connected. The evaluator shall ensure the message is encrypted and cannot be read in transit, and that the TOE is able to decrypt and read the message sent.

5.2.1.5 Cryptographic Key Support (REK)

FCS_CKM_EXT.1	**Extended: Cryptographic Key Support**

FCS_CKM_EXT.1.1: The TSF shall support a hardware-protected REK with an AES key of size [selection: *128 bits, 256 bits*].

FCS_CKM_EXT.1.2: A REK shall not be able to be read from or exported from the hardware.

FCS_CKM_EXT.1.3: System software on the TSF shall be able only to request encryption/decryption by the key and shall not be able to read, import, or export a REK.

FCS_CKM_EXT.1.4: A REK shall be generated by a RBG in accordance with **FCS_RBG_EXT.1**.

Application Note: Either 128-bit or 256-bit (or both) are allowed; the ST author makes the selection appropriate for the device.

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

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

Assurance Activity:

The evaluator shall review the TSS to determine that a REK is supported by the product, that the TSS includes a description of the protection provided by the product 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 actions are isolated so as to prevent applications and system-level processes from reading the REK while allowing encryption/decryption by the key.

The evaluator shall verify that the generation of a REK meets the **FCS_RBG_EXT.1.1** and **FCS_RBG_EXT.1.2** requirements:

- If REK(s) 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_EXT.1** is invoked, and whether a separate instance of the RBG is used for REK(s).
- If REK(s) is/are generated off-device, the TSS shall include evidence that the RBG meets **FCS_RBG_EXT.1.2**. This will likely a second set of RBG documentation equivalent to the documentation provided for the RBG assurance activities. In addition, the TSS shall describe the manufacturing process that prevents the device manufacturer from accessing any REKs.

5.2.1.6 Cryptographic Data Encryption Keys

FCS_CKM_EXT.2	Extended: Cryptographic Key Random Generation

FCS_CKM_EXT.2.1 All DEKs shall be randomly generated with entropy corresponding to the security strength of AES key sizes of [selection: *128, 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 a RBG implemented on the TOE device (**FCS_RBG_EXT.1**). Either 128-bit or 256-bit (or both) are allowed; the ST author makes the selection appropriate for the device. A DEK is used in addition to the KEK so that authentication factors (especially the Password Authentication Factor) can be changed without having to re-encrypt all of the user data on the device.

Assurance Activity:

The evaluator shall review the TSS to determine that it describes how the functionality described by **FCS_RBG_EXT.1** is invoked to generate DEKs. The evaluator uses the description of the RBG functionality in **FCS_RBG_EXT.1** or documentation available for the operational environment to determine that the key size being requested is identical to the key size and mode to be used for the encryption/decryption of the data.

5.2.1.7 Cryptographic Key Encryption Keys

FCS_CKM_EXT.3	Extended: Cryptographic Key Generation

FCS_CKM_EXT.3.1 All KEKs shall be [selection: *128-bit, 256-bit*] keys corresponding to at least the security strength of the keys encrypted by the KEK.

FCS_CKM_EXT.3.2 The TSF shall generate KEKs by deriving the KEK from a Password Authentication Factor using PBKDF and [selection:

*a) an RBG that meets this profile (as specified in **FCS_RBG_EXT.1**)*

b) combined from other KEKs in a way that preserves the effective entropy of each factor by [selection: using an XOR operation, concatenating the keys and use a KDF (as described in SP 800-108), encrypting one key with another]].

Application Note:

The PBKDF is performed in accordance with **FCS_COP.1(5)**.

It is expected that each of these methods will 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. If combined, the ST author shall describe which method of combination is used and shall justify that the effective entropy of each factor is preserved.

Assurance Activity:

The evaluator shall examine the password hierarchy TSS to ensure that the formation of all KEKs is described and that the key sizes match that described by the ST author.

- If the KEK is generated, the evaluator shall review the TSS to determine that it describes how the functionality described by **FCS_RBG_EXT.1** is invoked. The evaluator uses the description of the RBG functionality in **FCS_RBG_EXT.1** or documentation available for the operational environment to determine that the key size being requested is identical to the key size and mode to be used for the encryption/decryption of the data.
- If the KEK is formed from a combination, the evaluator shall verify that the TSS describes the method of combination and contains a justification for preserving the effective entropy.

5.2.1.8 Cryptographic Key Destruction

FCS_CKM_EXT.4	**Extended: Key Destruction**

FCS_CKM_EXT.4.1 The TSF shall destroy cryptographic keys in accordance with the specified cryptographic key destruction method [selection*:*

- *by clearing the KEK encrypting the target key,*
- *in accordance with the following rules:*
    - *For volatile 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_EXT.1), followed a read-verify.*
    - *For volatile flash memory the destruction shall be executed by* [*selection: a single direct overwrite consisting of zeros followed by a read-verify, a block erase followed by a read-verify.*]]

Application Note: The clearing indicated above applies to each intermediate storage area for plaintext key/cryptographic critical security parameter (i.e., any storage, such as memory buffers, that is included in the path of such data) upon the transfer of the key/cryptographic critical security parameter to another location.

FCS_CKM_EXT.4.2 The TSF shall destroy all plaintext keying material cryptographic security parameters when no longer needed.

Application Note: For the purposes of this requirement, plaintext keying material refers to authentication data, secret/private symmetric keys, data used to derive keys, etc. and does not refer to REKs. Key destruction procedures are performed in accordance with **FCS_CKM_EXT.4.1**.

There are multiple situations in which plaintext keying material is no longer necessary, including when the TOE is powered down, when the wipe function is performed, 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 that key material which is protected by the password-derived KEK according to **FCS_STG_EXT.2** – see Figure 3). In the future, "no longer needed" will include keys generated for protecting sensitive data received while in the locked state (**FDP_DAR_EXT.2**).

Assurance Activity:

The evaluator shall check to ensure the TSS describes when each of the plaintext keys (DEKs, software-based key storage, and KEKs) are cleared including on system power off, on wipe function, on disconnection of trusted channels, when no longer needed by the trusted channel per the protocol, when transitioning to the locked state (and possibly including immediately after use, while in the locked state, etc.); and the type of clearing procedure that is performed (cryptographic erase, 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 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 three times with a random pattern that is changed before each write").

Assurance 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.

For each key clearing situation, including on system power off, on wipe function, on disconnection of trusted channels, when no longer needed by the trusted channel per the protocol, and when transitioning to the locked state (and possibly including immediately after use, while in the locked state, etc.) the evaluator shall repeat the following test.

Test 1: The evaluator shall utilize appropriate combinations of specialized operational environment and development tools (debuggers, simulators, etc.) for the TOE and instrumented TOE builds to test that keys are cleared correctly, including *all* intermediate copies of the key that may have been created internally by the TOE during normal cryptographic processing with that key.

Cryptographic TOE implementations in software shall be loaded and exercised under a debugger to perform such tests. The evaluator shall perform the following test for each key subject to clearing, including intermediate copies of keys that are persisted encrypted by the TOE:

1. Load the instrumented TOE build in a debugger.
2. Record the value of the key in the TOE subject to clearing.
3. Cause the TOE to perform a normal cryptographic processing with the key from #1.
4. Cause the TOE to clear the key.
5. Cause the TOE to stop the execution but not exit.
6. Cause the TOE to dump the entire memory footprint of the TOE into a binary file.
7. Search the content of the binary file created in #4 for instances of the known key value from #1.

The test succeeds if no copies of the key from #1 are found in step #7 above and fails otherwise.

The evaluator shall perform this test on *all* keys, including those persisted in encrypted form, to ensure intermediate copies are cleared.

Test 2: In cases where the TOE is implemented in firmware and operates in a limited operating environment that does not allow the use of debuggers, the evaluator shall utilize a simulator for the TOE on a general purpose operating system. The evaluator shall provide a rationale explaining the instrumentation of the simulated test environment and justifying the obtained test results.

5.2.1.9 TSF Wipe

FCS_CKM_EXT.5	**Extended: TSF Wipe**

FCS_CKM_EXT.5.1 The TSF shall wipe all protected data by [selection*:*

Cryptographically erasing the encrypted DEKs and/or the KEKs in non-volatile memory by following the requirements in FCS_CKM_EXT.4.1;

Overwriting all protected data according to the following rules:

- *For EEPROM, whether volatile or non-volatile, 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_EXT.1, followed a read-verify.*
- *For flash memory, whether volatile or non-volatile, the destruction shall be executed [selection: by a single direct overwrite consisting of zeros followed by a read-verify, by a block erase followed by a read-verify]*.
- *For non-volatile memory other than EEPROM and flash, the destruction shall be executed by overwriting three or more times using a different alternating data pattern each time*.]

FCS_CKM_EXT.5.2 The TSF shall perform a power cycle on conclusion of the wipe procedure.

Application Note: The ST author shall select which method of wipe the TSF performs.

Assurance Activity:

The evaluator shall check to ensure the TSS describes how the device is wiped; and 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").

Assurance 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.

The assurance activities differ for the two wipe methods:

Test for Method 1: The evaluator shall enable encryption according to the AGD guidance. The evaluator shall use the test outlined for **FCS_CKM_EXT.4**, implementing the wipe command according to the AGD guidance provided for **FMT_SMF.1** and as defined in Test 1, Step 4 of the assurance activities specified following **FCS_CKM_EXT.4**.

Test for Method 2: The evaluator shall enable encryption according to the AGD guidance. The evaluator shall create user data (protected data), for example, by using an application. The evaluator shall use a tool provided by the developer to examine this data stored in memory. The evaluator shall initiate the wipe command according to the AGD guidance provided for **FMT_SMF.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. This test shall be repeated for each type of memory used to store the data to be protected.

5.2.1.10 Cryptographic Salt Generation

FCS_CKM_EXT.6	**Extended: Salt Generation**

FCS_CKM_EXT.6.1 The TSF shall generate all salts using a RBG that meets [**FCS_RBG_EXT.1**].

Assurance Activity: The ST author shall provide a description in the TSS regarding the salt generation. The evaluator shall confirm that the salt is generating using an RBG described in **FCS_RBG_EXT.1**.

5.2.2 Cryptographic Operation (FCS_COP)

5.2.2.1 Confidentiality Algorithms

FCS_COP.1(1)	Cryptographic operation

FCS_COP.1.1(1) The TSF shall perform [***encryption/decryption***] in accordance with a specified cryptographic algorithm

- AES-CBC (as defined in NIST SP 800-38A) mode,
- AES-CCMP (as defined in FIPS PUB 197, NIST SP 800-38C and IEEE 802.11-2012), and

[selection:

- *AES Key Wrap (KW) (as defined in NIST SP 800-38F), AES Key Wrap with Padding (KWP) (as defined in NIST SP 800-38F), AES-GCM (as defined in NIST SP 800-38D), AES-CCM (as defined in NIST SP 800-38C), AES-XTS (as defined in NIST SP 800-38E) mode, no other modes*]

and cryptographic key sizes 128-bit key sizes and [selection: *256-bit key sizes, no other key sizes*].

Application Note: For the first selection of **FCS_COP.1.1(1)**, the ST author should choose the mode or modes in which AES operates. For the second selection, the ST author should choose the key sizes that are supported by this functionality. 128-bit CBC and CCMP are required in order to comply with **FCS_TLS_EXT.1** and **FCS_CKM.1.1(3)**.

Note that to comply with IEEE 802.11-2012, AES CCMP (which uses AES in CCM as specified in SP 800-38C) with cryptographic key size of 128 bits must be implemented. In the future, as this standard is updated and new cryptographic modes are reviewed and approved by NIST, this requirement may include requirements for additional/new cryptographic modes and key sizes.

Assurance Activity:

AES-CBC Tests

AES-CBC Known Answer Tests

There are four Known Answer Tests (KATs), described below. In all KATs, the plaintext, ciphertext, and IV values shall be 128-bit blocks. 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.

> **KAT-1.** To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from AES-CBC encryption of the given plaintext using a key value of all zeros and an IV of all zeros. Five plaintext values shall be encrypted with a 128-bit all-zeros key, and the other five shall be encrypted with a 256-bit all-zeros key.

To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input and AES-CBC decryption.

KAT-2. To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 key values and obtain the ciphertext value that results from AES-CBC encryption of an all-zeros plaintext using the given key value and an IV of all zeros. Five of the keys shall be 128-bit keys, and the other five shall be 256-bit keys.

To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using an all-zero ciphertext value as input and AES-CBC decryption.

KAT-3. To test the encrypt functionality of AES-CBC, the evaluator shall supply the two sets of key values described below and obtain the ciphertext value that results from AES encryption of an all-zeros plaintext using the given key value and an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second set shall have 256 256-bit keys. Key *i* in each set shall have the leftmost *i* bits be ones and the rightmost *N-i* bits be zeros, for *i* in [1,N].

To test the decrypt functionality of AES-CBC, the evaluator shall supply the two sets of key and ciphertext value pairs described below and obtain the plaintext value that results from AES-CBC decryption of the given ciphertext using the given key and an IV of all zeros. The first set of key/ciphertext pairs shall have 128 128-bit key/ciphertext pairs, and the second set of key/ciphertext pairs shall have 256 256-bit key/ciphertext pairs. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N]. The ciphertext value in each pair shall be the value that results in an all-zeros plaintext when decrypted with its corresponding key.

KAT-4. To test the encrypt functionality of AES-CBC, the evaluator shall supply the set of 128 plaintext values described below and obtain the two ciphertext values that result from AES-CBC encryption of the given plaintext using a 128-bit key value of all zeros with an IV of all zeros and using a 256-bit key value of all zeros with an IV of all zeros, respectively. Plaintext value i in each set shall have the leftmost i bits be ones and the rightmost 128-i bits be zeros, for i in [1,128].

To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using ciphertext values of the same form as the plaintext in the encrypt test as input and AES-CBC decryption.

AES-CBC Multi-Block Message Test

The evaluator shall test the encrypt functionality by encrypting an *i*-block message where 1 < *i* <=10. The evaluator shall choose a key, an IV and plaintext message of length *i* blocks and encrypt the message, using the mode to be tested, with the chosen key and IV. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key and IV using a known good implementation.

The evaluator shall also test the decrypt functionality for each mode by decrypting an *i*-block message where 1 < *i* <=10. The evaluator shall choose a key, an IV and a ciphertext message of length *i* blocks and decrypt the message, using the mode to be tested, with the chosen key and IV. The plaintext shall be compared to the result of decrypting the same ciphertext message with the same key and IV using a known good implementation.

AES-CBC Monte Carlo Tests

The evaluator shall test the encrypt functionality using a set of 200 plaintext, IV, and key 3-tuples. 100 of these shall use 128 bit keys, and 100 shall use 256 bit keys. The plaintext and IV values shall be 128-bit blocks. For each 3-tuple, 1000 iterations shall be run as follows:

```
# Input: PT, IV, Key
for i = 1 to 1000:
        if i == 1:
                CT[1] = AES-CBC-Encrypt(Key, IV, PT)
                PT = IV
        else:
                CT[i] = AES-CBC-Encrypt(Key, PT)
                PT = CT[i-1]
```

The ciphertext computed in the 1000th iteration (i.e., CT[1000]) is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation.

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

AES-CCM Tests

The evaluator shall test the generation-encryption and decryption-verification functionality of AES-CCM for the following input parameter and tag lengths:

128 bit and 256 bit keys

Two payload lengths. One payload length shall be the shortest supported payload length, greater than or equal to zero bytes. The other payload length shall be the longest supported payload length, less than or equal to 32 bytes (256 bits).

Two or three associated data lengths. One associated data length shall be 0, if supported. One associated data length shall be the shortest supported payload length, greater than or equal to zero bytes. One associated data length shall be the longest supported payload length, less than or equal to 32 bytes (256 bits). If the implementation supports an associated data length of 2^{16} bytes, an associated data length of 2^{16} bytes shall be tested.

Nonce lengths. All supported nonce lengths between 7 and 13 bytes, inclusive, shall be tested.

Tag lengths. All supported tag lengths of 4, 6, 8, 10, 12, 14 and 16 bytes shall be tested.

To test the generation-encryption functionality of AES-CBC, the evaluator shall perform the following four tests:

Test 1. For EACH supported key and associated data length and ANY supported payload, nonce and tag length, the evaluator shall supply one key value, one nonce value and 10 pairs of associated data and payload values and obtain the resulting ciphertext.

Test 2. For EACH supported key and payload length and ANY supported associated data, nonce and tag length, the evaluator shall supply one key value, one nonce value and 10 pairs of associated data and payload values and obtain the resulting ciphertext.

Test 3. For EACH supported key and nonce length and ANY supported associated data, payload and tag length, the evaluator shall supply one key value and 10 associated data, payload and nonce value 3-tuples and obtain the resulting ciphertext.

Test 4. For EACH supported key and tag length and ANY supported associated data, payload and nonce length, the evaluator shall supply one key value, one nonce value and 10 pairs of associated data and payload values and obtain the resulting ciphertext.

To determine correctness in each of the above tests, the evaluator shall compare the ciphertext with the result of generation-encryption of the same inputs with a known good implementation.

To test the decryption-verification functionality of AES-CBC, for EACH combination of supported associated data length, payload length, nonce length and tag length, the evaluator shall supply a key value and 15 nonce, associated data and ciphertext 3-tuples and obtain either a FAIL result or a PASS result with the decrypted payload. The evaluator shall supply 10 tuples that should FAIL and 5 that should PASS per set of 15.

Additionally, the evaluator shall use tests from the IEEE 802.11-02/362r6 document "Proposed Test vectors for IEEE 802.11 TGi", dated September 10, 2002, Section 2.1 AES-CCMP Encapsulation Example and Section 2.2 Additional AES CCMP Test Vectors to further verify the IEEE 802.11-2007 implementation of AES-CCMP.

AES-GCM Monte Carlo Test

The evaluator shall test the authenticated encrypt functionality of AES-GCM for each combination of the following input parameter lengths:

128 bit and 256 bit keys

Two plaintext lengths. One of the plaintext lengths shall be a non-zero integer multiple of 128 bits, if supported. The other plaintext length shall not be an integer multiple of 128 bits, if supported.

Three AAD lengths. One AAD length shall be 0, if supported. One AAD length shall be a non-zero integer multiple of 128 bits, if supported. One AAD length shall not be an integer multiple of 128 bits, if supported.

Two IV lengths. If 96 bit IV is supported, 96 bits shall be one of the two IV lengths tested.

The evaluator shall test the encrypt functionality using a set of 10 key, plaintext, AAD, and IV tuples for each combination of parameter lengths above and obtain the ciphertext value and tag that results from AES-GCM authenticated encrypt. Each supported tag length shall be tested at least once per set of 10. The IV value may be supplied by the evaluator or the implementation being tested, as long as it is known.

The evaluator shall test the decrypt functionality using a set of 10 key, ciphertext, tag, AAD, and IV 5-tuples for each combination of parameter lengths above and obtain a Pass/Fail result on authentication and the decrypted plaintext if Pass. The set shall include five tuples that Pass and five that Fail.

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.

XTS-AES Monte Carlo Test

The evaluator shall test the encrypt functionality of XTS-AES for each combination of the following input parameter lengths:

256 bit (for AES-128) and 512 bit (for AES-256) keys

Three data unit (i.e., plaintext) lengths. One of the data unit lengths shall be a non-zero integer multiple of 128 bits, if supported. One of the data unit lengths shall be an integer multiple of 128 bits, if supported. The third data unit length shall be either the longest supported data unit length or 2^{16} bits, whichever is smaller.

using a set of 100 (key, plaintext and 128-bit random tweak value) 3-tuples and obtain the ciphertext that results from XTS-AES encrypt.

The evaluator may supply a data unit sequence number instead of the tweak value if the implementation supports it. The data unit sequence number is a base-10 number ranging between 0 and 255 that implementations convert to a tweak value internally.

The evaluator shall test the decrypt functionality of XTS-AES using the same test as for encrypt, replacing plaintext values with ciphertext values and XTS-AES encrypt with XTS-AES decrypt.

AES Key Wrap (AES-KW) and Key Wrap with Padding (AES-KWP) Test

The evaluator shall test the authenticated encryption functionality of AES-KW for EACH combination of the following input parameter lengths:

128 and 256 bit key encryption keys (KEKs)

Three plaintext lengths. One of the plaintext lengths shall be two semi-blocks (128 bits). One of the plaintext lengths shall be three semi-blocks (192 bits). The third data unit length shall be the longest supported plaintext length less than or equal to 64 semi-blocks (4096 bits).

using a set of 100 key and plaintext pairs and obtain the ciphertext that results from AES-KW authenticated encryption. To determine correctness, the evaluator shall use the AES-KW authenticated-encryption function of a known good implementation.

The evaluator shall test the authenticated-decryption functionality of AES-KW using the same test as for authenticated-encryption, replacing plaintext values with ciphertext values and AES-KW authenticated-encryption with AES-KW authenticated-decryption.

The evaluator shall test the authenticated-encryption functionality of AES-KWP using the same test as for AES-KW authenticated-encryption with the following change in the three plaintext lengths:

> One plaintext length shall be one octet. One plaintext length shall be 20 octets (160 bits).
>
> One plaintext length shall be the longest supported plaintext length less than or equal to 512 octets (4096 bits).

The evaluator shall test the authenticated-decryption functionality of AES-KWP using the same test as for AES-KWP authenticated-encryption, replacing plaintext values with ciphertext values and AES-KWP authenticated-encryption with AES-KWP authenticated-decryption.

5.2.2.2 Hashing Algorithms

FCS_COP.1(2)	**Cryptographic operation**

FCS_COP.1.1(2) The TSF shall perform [***cryptographic hashing***] in accordance with a specified cryptographic algorithm SHA-1 and [selection*: SHA-256, SHA-384, SHA-512, no other algorithms*] and message digest sizes 160 and [selection*: 256, 384, 512 bits, no other message digest sizes*] that meet the following: [***FIPS Pub 180-4***].

Application Note: In future versions of this document, SHA-1 may be removed as an option. SHA-1 for generating digital signatures will no longer be allowed after December 2013, and SHA-1 for verification of digital signatures is strongly discouraged as there may be risk in accepting these signatures. SHA-1 is currently required in order to comply with **FCS_TLS_EXT.1** and **FCS_CKM.1.1(3)**.

The intent of this requirement is to specify the hashing function. 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 example, SHA 256 for 128-bit keys).

Assurance Activity:
The evaluator checks the AGD documents to determine that any configuration that is required to be done to configure the functionality for the required hash sizes is present. The evaluator shall check that the association of the hash function with other TSF cryptographic functions (for example, the digital signature verification function) is documented in the TSS.

The TSF hashing functions can be implemented in one of two modes. The first mode is the byte-oriented mode. In this mode the TSF only hashes messages that are an integral number

of bytes in length; i.e., the length (in bits) of the message to be hashed is divisible by 8. The second mode is the bit-oriented mode. In this mode the TSF hashes messages of arbitrary length. As there are different tests for each mode, an indication is given in the following sections for the bit-oriented vs. the byte-oriented testmacs.

The evaluator shall perform all of the following tests for each hash algorithm implemented by the TSF and used to satisfy the requirements of this PP.

Short Messages Test - Bit-oriented Mode
The evaluators devise an input set consisting of m+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m bits. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Short Messages Test - Byte-oriented Mode
The evaluators devise an input set consisting of m/8+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m/8 bytes, with each message being an integral number of bytes. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Selected Long Messages Test - Bit-oriented Mode
The evaluators devise an input set consisting of m messages, where m is the block length of the hash algorithm. The length of the ith message is 512 + 99*i, where $1 \leq i \leq m$. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Selected Long Messages Test - Byte-oriented Mode
The evaluators devise an input set consisting of m/8 messages, where m is the block length of the hash algorithm. The length of the ith message is 512 + 8*99*i, where $1 \leq i \leq m/8$. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Pseudorandomly Generated Messages Test
This test is for byte-oriented implementations only. The evaluators randomly generate a seed that is n bits long, where n is the length of the message digest produced by the hash function to be tested. The evaluators then formulate a set of 100 messages and associated digests by following the algorithm provided in Figure 1 of [SHAVS]. The evaluators then ensure that the correct result is produced when the messages are provided to the TSF.

5.2.2.3 Signature Algorithms

FCS_COP.1(3)	Refinement: Cryptographic operation

FCS_COP.1.1(3) The TSF shall perform [***cryptographic signature services (generation and verification)***] in accordance with a specified cryptographic algorithm

- FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3 for RSA schemes

[selection:

- *FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4 for ECDSA schemes and implementing "NIST curves" P-256, P-384 and [selection: P-521, no other curves*
- *No other algorithms*].

and cryptographic key sizes [***equivalent to, or greater than, a symmetric key strength of 112 bits***].

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/selections to specify the parameters that are implemented for that algorithm. RSA signature generation and verification is currently required in order to comply with **FCS_TLS_EXT.1**.

Assurance Activity:
Key Generation:
Key Generation for RSA Signature Schemes

The evaluator shall verify the implementation of RSA Key Generation by the TOE using the Key Generation test. This test verifies the ability of the TSF to correctly produce values for the key components including the public verification exponent *e*, the private prime factors *p* and *q*, the public modulus *n* and the calculation of the private signature exponent *d*.

Key Pair generation specifies 5 ways (or methods) to generate the primes *p* and *q*. These include:

1. Random Primes:
   - Provable primes
   - Probable primes
2. Primes with Conditions:
   - Primes *p1, p2, q1,q2, p* and *q* shall all be provable primes
   - Primes *p1, p2, q1,* and *q2* shall be provable primes and *p* and *q* shall be probable primes
   - Primes *p1, p2, q1,q2, p* and *q* shall all be probable primes

To test the key generation method for the Random Provable primes method and for all the Primes with Conditions methods, the evaluator must seed the TSF key generation routine with sufficient data to deterministically generate the RSA key pair. This includes the random seed(s), the public exponent of the RSA key, and the desired key length. For each key length supported, 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 from a known good implementation.

ECDSA Key Generation Tests

FIPS 186-4 ECDSA Key Generation Test

For each supported NIST curve, i.e., P-256, P-284 and P-521, the evaluator shall require the implementation under test (IUT) to generate 10 private/public key pairs. The private key shall be generated using an approved random bit generator (RBG). To determine correctness, the evaluator shall submit the generated key pairs to the public key verification (PKV) function of a known good implementation.

FIPS 186-4 Public Key Verification (PKV) Test

For each supported NIST curve, i.e., P-256, P-284 and P-521, the evaluator shall generate 10 private/public key pairs using the key generation function of a known good implementation and modify five of the public key values so that they are incorrect, leaving five values unchanged (i.e., correct). The evaluator shall obtain in response a set of 10 PASS/FAIL values.

ECDSA Algorithm Tests

CDSA FIPS 186-4 Signature Generation Test

For each supported NIST curve (i.e., P-256, P-284 and P-521) and SHA function pair, the evaluator shall generate 10 1024-bit long messages and obtain for each message a public key and the resulting signature values R and S. To determine correctness, the evaluator shall use the signature verification function of a known good implementation.

ECDSA FIPS 186-4 Signature Verification Test

For each supported NIST curve (i.e., P-256, P-284 and P-521) and SHA function pair, the evaluator shall generate a set of 10 1024-bit message, public key and signature tuples and modify one of the values (message, public key or signature) in five of the 10 tuples. The evaluator shall obtain in response a set of 10 PASS/FAIL values.

RSA Signature Algorithm Tests

Signature Generation Test

The evaluator shall verify the implementation of RSA Signature Generation by the TOE using the Signature Generation Test. To conduct this test the evaluator must generate or obtain 10 messages from a trusted reference implementation for each modulus size/SHA combination supported by the TSF. The evaluator shall have the TOE use their private key and modulus value to sign these messages.

The evaluator shall verify the correctness of the TSF's signature using a known good implementation and the associated public keys to verify the signatures.

Signature Verification Test
The evaluator shall perform the Signature Verification test to verify the ability of the TOE to recognize another party's valid and invalid signatures. The evaluator shall inject errors into the test vectors produced during the Signature Verification Test by introducing errors in some of the public keys e, messages, IR format, and/or signatures. The TOE attempts to verify the signatures and returns success or failure.

The evaluator shall use these test vectors to emulate the signature verification test using the corresponding parameters and verify that the TOE detects these errors.

5.2.2.4 Keyed Hash Algorithms

FCS_COP.1(4)	**Refinement: Cryptographic operation**

FCS_COP.1.1(4) The TSF shall perform [***keyed-hash message authentication***] in accordance with a specified cryptographic algorithm HMAC-SHA-1 and [*selection: HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512, no other algorithms*] and cryptographic key sizes [assignment: *key size (in bits) used in HMAC*] and message digest sizes 160 and [selection: *256, 384, 512, no other*] bits that meet the following: [***FIPS Pub 198-1, "The Keyed-Hash Message Authentication Code, and FIPS Pub 180-4, "Secure Hash Standard***].

Application Note: The selection in this requirement must be consistent with the key size specified for the size of the keys used in conjunction with the keyed-hash message authentication. HMAC-SHA-1 is currently required in order to comply with **FCS_TLS_EXT.1** and **FCS_CKM.1.1(3)** but may be removed in future versions of this document.

Assurance Activity:
The evaluator shall examine the TSS to ensure that it specifies the following values used by the HMAC function: key length, hash function used, block size, and output MAC length used.

For each of the supported parameter sets, the evaluator shall compose 15 sets of test data. Each set shall consist of a key and message data. The evaluator shall have the TSF generate HMAC tags for these sets of test data. The resulting MAC tags shall be compared to the result of generating HMAC tags with the same key and IV using a known good implementation.

5.2.2.5 Password-Based Key Derivation Functions

FCS_COP.1(5)	**Refinement: Cryptographic operation**

FCS_COP.1.1(5) The TSF shall perform [***Password-based Key Derivation Functions***] in accordance with a specified cryptographic algorithm [*HMAC-[selection: SHA-1, SHA-256, SHA-384, SHA-512]*] and output cryptographic key sizes [selection: *128, 256*] that meet the following: [NIST SP 800-132].

Application Note: The key cryptographic key sizes in the second selection should be made to correspond to the KEK key sizes selected in **FCS_CKM_EXT.3**. A future requirement will require a PBKDF iteration count of at least 1000.

This password must be conditioned into a string of bits that forms the submask to be used as input into the KEK. Conditioning can be performed using one of the identified hash functions or the process described in NIST SP 800-132; the method used is selected by the ST Author. 800-132 requires the use of a pseudo-random function (PRF) consisting of HMAC with an approved hash function. The ST author selects the hash function used, also includes the appropriate requirements for HMAC and the hash function.

Assurance Activity:

The evaluator shall check that the TSS describes the method by which the password is first encoded and then fed to the SHA algorithm. The settings for the algorithm (padding, blocking, etc.) shall be described, and the evaluator shall verify that these are supported by the selections in this component as well as the selections concerning the hash function itself. The evaluator shall verify that the TSS contains a description of how the output of the hash function is used to form the submask that will be input into the function and is the same length as the DEK as specified in **FCS_CKM_EXT.2**.

For the NIST SP 800-132-based conditioning of the passphrase, the required assurance activities will be performed when doing the assurance activities for the appropriate requirements (FCS_COP.1.1(4)). If any manipulation of the key is performed in forming the submask that will be used to form the KEK, that process shall be described in the TSS.

No explicit testing of the formation of the submask from the input password is required.

5.2.3 Initialization Vector Generation (FCS_IV)

FCS_IV_EXT.1	**Extended: Initialization Vector Generation**

FCS_IV_EXT.1.1 The TSF shall generate IVs in accordance with Table 11: References and IV Requirements for NIST-approved Cipher Modes.

Application Note: Table 11 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.

Assurance Activity:

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**.

5.2.4 Random Bit Generation (FCS_RBG)

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

FCS_RBG_EXT.1.1: The TSF shall perform all deterministic random bit generation services in accordance with [selection, *choose one of: NIST Special Publication 800-90A using* [selection: *Hash_DRBG (any), HMAC_DRBG (any), CTR_DRBG (AES), Dual_EC_DRBG (any)]; FIPS Pub 140-2 Annex C: X9.31 Appendix 2.4 using AES*].

FCS_RBG_EXT.1.2 The deterministic RBG shall be seeded by an entropy source that accumulates entropy from [selection: *a software-based noise source, TSF-hardware-based noise source*] with a minimum of [selection: *128 bits, 256 bits*] of entropy at least equal to the greatest security strength (according to NIST SP 800-57) of the keys and hashes that it will generate.

FCS_RBG_EXT.1.3: The TSF shall be capable of providing output of the RBG to applications running on the TSF that request random bits.

Application Note: NIST Special Pub 800-90B, Appendix C describes the minimum entropy measurement that should be used immediately and will be required in future publications of this PP.

For the first selection in FCS_RBG_EXT.1.1, the ST author should select the standard to which the RBG services comply (either SP 800-90A or FIPS 140-2 Annex C).

SP 800-90A contains four 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 (if SP 800-90A is selected), and include the specific underlying cryptographic primitives used in the requirement or in the TSS. While any of the identified hash functions (SHA-1, SHA-224, SHA-256, SHA-384, SHA-512) are allowed for Hash_DRBG or HMAC_DRBG, only AES-based implementations for CTR_DRBG are allowed. While any of the curves defined in SP800-90A are allowed for Dual_EC_DRBG, the ST author not only must include the curve chosen, but also the hash algorithm used.

Note that for FIPS Pub 140-2 Annex C, currently only the method described in NIST-Recommended Random Number Generator Based on ANSI X9.31 Appendix A.2.4 Using the 3-Key Triple DES and AES Algorithms, Section 3 is valid. If the key length for the AES implementation used here is different than that used to encrypt the user data, then FCS_COP.1 may have to be adjusted or iterated to reflect the different key length. For the selection in FCS_RBG_EXT.1.2, the ST author selects the minimum number of bits of entropy that is used to seed the RBG.

The ST author must also ensure that any underlying functions are included in the baseline requirements for the TOE.

For the selection in FCS_RBG_EXT.1.2, the ST author selects the appropriate number of bits of entropy that corresponds to the greatest security strength of the algorithms included in the ST. Security strength is defined in Tables 2 and 3 of NIST SP 800-57A. For example, if the implementation includes 2048-bit RSA (security strength of 112 bits), AES 128 (security strength 128 bits), and HMAC-512 (security strength 256 bits), then the ST author would select 256 bits.

In the future, this profile will require at least one hardware-based noise source; the ST author may select additional noise source. A hardware noise source is a component that produces data that cannot be explained by a deterministic rule, due to its physical nature. In other words, a hardware based noise source generates sequences of random numbers from a physical process that cannot be predicted. For example, a sampled ring oscillator consists of an odd number of inverter gates chained into a loop, with an electrical pulse traveling from inverter to inverter around the loop. The inverters are not clocked, so the precise time required for a complete circuit around the loop varies slightly as various physical effects

modify the small delay time at each inverter on the line to the next inverter. This variance results in an approximate natural frequency that contains drift and jitter over time. The output of the ring oscillator consists of the oscillating binary value sampled at a constant rate from one of the inverters – a rate that is significantly slower than the oscillator's natural frequency.

Any hardware component behaving in similarly variable ways that cannot be explained by a precise and predictable rule can serve as a hardware-based noise source. It is also possible to use multiple independent noise sources to increase entropy production and reduce attack potential (by requiring attackers to exploit multiple random bit streams) as long as at least one of the sources is hardware based. It should be noted that timing of interrupts caused by mechanical I/O devices and system counters are not considered *hardware-based* noise sources for the purposes of this requirement.

Assurance Activity:

Documentation shall be produced—and the evaluator shall perform the activities—in accordance with Annex E.

The evaluator shall verify that the API documentation provided according to Section 6.2.1 includes the security functions described in **FCS_RBG_EXT.1.3**.

The evaluator shall perform the following tests, depending on the standard to which the RBG conforms.

Implementations Conforming to FIP 140-2 Annex C

The reference for the tests contained in this section is The Random Number Generator Validation System (RNGVS). The evaluators shall conduct the following two tests. Note that the "expected values" are produced by a reference implementation of the algorithm that is known to be correct. Proof of correctness is left to each Scheme.

The evaluators shall perform a Variable Seed Test. The evaluators shall provide a set of 128 (Seed, DT) pairs to the TSF RBG function, each 128 bits. The evaluators shall also provide a key (of the length appropriate to the AES algorithm) that is constant for all 128 (Seed, DT) pairs. The DT value is incremented by 1 for each set. The seed values shall have no repeats within the set. The evaluators ensure that the values returned by the TSF match the expected values.

The evaluators shall perform a Monte Carlo Test. For this test, they supply an initial Seed and DT value to the TSF RBG function; each of these is 128 bits. The evaluators shall also provide a key (of the length appropriate to the AES algorithm) that is constant throughout the test. The evaluators then invoke the TSF RBG 10,000 times, with the DT value being incremented by 1 on each iteration, and the new seed for the subsequent iteration produced as specified in NIST-Recommended Random Number Generator Based on ANSI X9.31 Appendix A.2.4 Using the 3-Key Triple DES and AES Algorithms, Section 3. The evaluators ensure that the 10,000th value produced matches the expected value.

Implementations Conforming to NIST Special Publication 800-90A

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 SP800-90A).

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 paragraphs contain 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 <= 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.

5.2.5 Cryptographic Algorithm Services (FCS_SRV)

FCS_SRV_EXT.1	**Extended: 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:

- FCS_COP.1(1)
- FCS_COP.1(3)
- FCS_COP.1(2)
- FCS_COP.1(4)
- FCS_COP.1(5)

- FCS_CKM.1(1)

[selection:

- FCS_CKM.1(2),
- No other cryptographic operations].

Assurance Activity:

The evaluator shall verify that the API documentation provided according to Section 6.2.1 includes the security functions (cryptographic algorithms) described in these requirements.

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 validation match the expected results according to the API documentation. This application may be used to assist in verifying the cryptographic operation assurance activities for the other algorithm services requirements.

5.2.6 Cryptographic Key Storage (FCS_STG)

This section describes how keys are protected. All keys must ultimately be protected by a REK, and may optionally be protected by the user's password. Each key's confidentiality and integrity must be protected. This section also describes the secure key storage services to be provided by the Mobile Device for use by applications and users, applying the same level of protection for these keys as keys internal to the OS.

5.2.6.1 Secure Key Storage

FCS_STG_EXT.1	Extended: Cryptographic Key Storage

FCS_STG_EXT.1.1 The TSF shall provide secure key storage for asymmetric private keys and [selection: *symmetric keys, persistent secrets, no other keys*].

Application Note: This secure key storage may be implemented fully in hardware or in software that is protected as required by **FCS_STG_EXT.2**. Support for secure key storage for symmetric keys, persistent secrets, and ECDSA will be tested in future revisions.

FCS_STG_EXT.1.2 The TSF shall be capable of importing keys/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 Author selects only user, the ST Author must also select function 14 in **FMT_MOF.1.1(1)**.

FCS_STG_EXT.1.3 The TSF shall be capable of destroying keys/secrets in the secure key storage upon request of [selection: *the user, the administrator*].

Application Note: If the ST Author selects only user, the ST Author must also select function 15 in **FMT_MOF.1.1(1)**.

FCS_STG_EXT.1.4 The TSF shall have the capability to allow only the application that imported the key/secret the use of the key/secret. Exceptions may only be explicitly authorized by [selection: *the user, the administrator, a common application developer*].

Application Note: If the ST Author selects user or administrator, the ST Author must also select function 39 in **FMT_SMF.1.1**. If the ST Author selects only user, the ST Author must also select function 25 in **FMT_MOF.1.1(1)**.

FCS_STG_EXT.1.5 The TSF shall allow only the application that imported the key/secret to request that the key/secret be destroyed. Exceptions may only be explicitly authorized by [selection: *the user, the administrator, a common application developer*].

Application Note: If the ST Author selects user or administrator, the ST Author must also select function 40 in **FMT_SMF.1.1**. If the ST Author selects only user, the ST Author must also select function 26 in **FMT_MOF.1.1(1)**.

Assurance Activity:

The assurance activity for this component entails examination of the ST's TSS to determine that the TOE's implements the required secure key storage.

The evaluator shall review the AGD guidance to determine that it describes the steps needed to import or destroy keys/secrets. The evaluator shall also verify that the API documentation provided according to Section 6.2.1 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/secrets in order to meet **FCS_STG_EXT.1.4**.

The evaluator shall test the functionality of each security function:

Test 1: The evaluator shall import keys/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/secret of each supported type and calls the import functions. The evaluator shall verify that no errors occur during import.

Test 2: The evaluator shall write, or the developer shall provide access to, an application that uses an imported key/secret:

- For RSA, the secret shall be used to sign data.

In the future additional types will be required to be tested:

- For ECDSA, the secret shall be used to sign data
- 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/secrets and a different application's imported keys/secrets. The evaluator shall verify that the TOE requires approval before allowing the application to use the key/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/secret as described.

- The evaluator shall repeat the test, allowing the approvals to verify that the application is able to use the key/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 3: The evaluator shall destroy keys/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/secret.

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

Assurance 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 5: The evaluator shall enable encryption according to the AGD guidance. The evaluator shall use the test outlined for **FCS_CKM_EXT.4**, destroy keys/secrets according to the AGD guidance provided for **FMT_SMF_EXT.1** and as defined in Test 1, Step 4 of the assurance activities specified following **FCS_CKM_EXT.4.**

5.2.6.2 Encryption of Stored Keys

FCS_STG_EXT.2	**Extended: Encrypted Cryptographic Key Storage**

FCS_STG_EXT.2.1 The TSF shall encrypt all DEKs and KEKs and [*selection: all software-based key storage, no other keys*] by KEKs that are [*selection:*

1) *Protected by the REK with [selection:*
    a. *encryption by a REK,*
    b. *encryption by a KEK chaining to a REK],*
2) *Protected by the REK and the password with [selection:*
    a. *encryption by a REK and the password-derived KEK,*
    b. *encryption by a KEK chaining to a REK and the password-derived KEK]*

].

Application Notes:

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.

Sensitive data shall be protected by the REK and the password. If **FDP_DAR_EXT.2** and **FCS_DAR_EXT.3** are included in the main body, this sensitive data includes some or all user or enterprise data. The software-based key storage shall either all be sensitive (protected by the REK and the password) or shall allow users and applications to mark the key as sensitive (protected by the REK and the password) according to **FDP_DAR_EXT.2.1**.

All keys must ultimately be protected by the REK. Sensitive data must be protected by the password (selection 2). In particular, Figure 3 has KEKs protected according to these requirements: DEK_1 meets 2a and would be appropriate for sensitive data, DEK_2 meets 1b and would not be appropriate for sensitive data, K_1 meets 1a and is not considered a sensitive key, and K_2 meets 2b and is considered a sensitive key.

Assurance Activity:

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, and of KEK related to the protection of the DEKs 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_EXT.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 KEY-4), 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.

FCS_STG_EXT.2.2 All keys shall be encrypted using AES in the [selection: *Key Wrap (KW) mode, Key Wrap with Padding (KWP) mode, GCM, CCM, CBC mode*].

Application Note: Either 128-bit or 256-bit (or both) are allowed; the ST author makes the selection appropriate for the device. This requirement refers only to KEKs as defined this PP and does not refer to those KEKs specified in other standards.

Assurance Activity:

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 modes.

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**.

5.2.6.3 Integrity of Stored Keys

FCS_STG_EXT.3	**Extended: Integrity of encrypted key storage**

FCS_STG_EXT.3.1 The TSF shall protect the integrity of any encrypted KEK 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(2)****) of the stored key that is encrypted by a key protected by* ***FCS_STG_EXT.2****;*
- *a keyed hash (****FCS_COP.1(4)****) 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***].

FCS_STG_EXT.3.2 The TSF shall verify the integrity of the [selection***:*** *hash, digital signature*] of the stored key prior to use of the key.

Application Note: 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 Assurance Activities for each of the options will be addressed in each of the requirements (**FCS_COP.1.1(2), FCS_COP.1(4)**, **FCS_CKM.1**).

Assurance Activity:

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**.

5.2.7 TLS Protocol (FCS_TLS)

5.2.7.1 EAP-TLS Protocol

FCS_TLS_EXT.1	**Extended: EAP TLS Protocol**

FCS_TLS_EXT.1.1 The TSF shall implement the EAP-TLS protocol as specified in RFC 5216 implementing TLS 1.0 (RFC 2246) and [selection: *TLS 1.1 (RFC 4346), TLS 1.2 (RFC 5246), no other TLS versions*] supporting the following ciphersuites: [

- *Mandatory Ciphersuites in accordance with RFC 3268:*
  - *TLS_RSA_WITH_AES_128_CBC_SHA*
- *[selection: Optional Ciphersuites:*
  - *TLS_RSA_WITH_AES_256_CBC_SHA*
  - *TLS_DHE_RSA_WITH_AES_128_CBC_SHA*
  - *TLS_DHE_RSA_WITH_AES_256_CBC_SHA*
  - *TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246*
  - *TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246*
  - *TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246*
  - *TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246*
  - *TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289*

- *TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289*
- *TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 6460*
- *TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 6460*
- [*assignment: Any other supported cipher suites*], *no other ciphersuite*]]

Application Note: The ciphersuites to be tested in the evaluated configuration are limited by this requirement. The ST author should select the optional ciphersuites that are supported; if there are no ciphersuites supported other than the mandatory suites, then "None" should be selected. It is necessary to limit the ciphersuites that can be used in an evaluated configuration administratively on the server in the test environment.

TLS 1.2 is the preferred protocol and may be required for EAP-TLS in the future; however, TLS 1.0 is currently required in order to ensure compliance with RFC 5216. TLS 1.2 is required for **FCS_TLS_EXT.2.** The Suite B algorithms listed above (RFC 6460) are the preferred algorithms for implementation.

FCS_TLS_EXT.1.2 The TSF shall verify that the server certificate presented for EAP-TLS [selection: *chains to one of the specified CAs, contains the specified FQDN of the acceptable authentication server certificate*.].

Application Note: The CA or FQDN is specified according to **FMT_SMF.1** function 7a.

Assurance Activity:

The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the ciphersuites supported are specified. The evaluator shall check the TSS to ensure that the ciphersuites specified are identical to those listed for this component. The evaluator shall also check the operational guidance to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS (for instance, the set of ciphersuites advertised by the TOE may have to be restricted to meet the requirements).

The evaluator shall check that the AGD guidance contains instructions for the administrator to configure the list of Certificate Authorities that are allowed to sign certificates or to configure the FQDN of the authentication server certificate that will be accepted by the TOE in the EAP-TLS exchange.

Additional tests may be added in the future to test compliance with RFC 5246. The evaluator shall also perform the following tests:

- *Test 1:* The evaluator shall establish a TLS connection using each of the ciphersuites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an EAP session. It is sufficient to observe the successful negotiation of a ciphersuite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic in an attempt to discern the ciphersuite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).
- *Test 2:* The following test is repeated for each supported certificate signing algorithm supported. The evaluator shall attempt to establish the connection using a server with a authentication server certificate that contains the Server Authentication purpose in

the extendedKeyUsage field and verify that a connection is established. The evaluator will then verify that the client rejects an otherwise valid server certificate that lacks the Server Authentication purpose in the extendedKeyUsage field and a connection is not established. Ideally, the two certificates should be identical except for the extendedKeyUsage field.

- *Test 3:* Following the guidance provided by the AGD guidance, a CA or an FQDN will be configured as "acceptable" for authentication server certificates and then the evaluator will start a wireless connection and verify that the wireless client is able to successfully connect. The evaluator will then configure the system such that an otherwise valid certificate is signed by a CA that is not allowed by the TOE or presents a FQDN that is not allowed by the TOE. Attempts to authenticate to an authentication server presenting such a certificate should result in the connection being refused. If the TOE supports both methods of limiting the acceptable authentication servers, the evaluator shall repeat this test twice, once with each method.

- *Test 4:* The evaluator shall configure the authentication server to send a certificate in the TLS connection that the does not match the server-selected ciphersuite (for example, send a ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite or send a RSA certificate while using one of the ECDSA ciphersuites.) The evaluator shall verify that the TOE disconnects after receiving the server's Certificate handshake message.

- *Test 5*: The evaluator shall setup a man-in-the-middle tool between the TOE and the authentication server and shall perform the following modifications to the traffic:

    ○ Modify at least one byte in the server's nonce in the Server Hello handshake message, and verify that the server denies the client's Finished handshake message.

    ○ Modify the server's selected ciphersuite in the Server Hello handshake message to be a ciphersuite not presented in the Client Hello handshake message. The evaluator shall verify that the client rejects the connection after receiving the Server Hello.

    ○ *(conditional)* If a DHE or ECDHE ciphersuite is supported, modify the signature block in the Server's KeyExchange handshake message, and verify that the client rejects the connection after receiving the Server KeyExchange.

    ○ Modify a byte in a CA field in the Server's Certificate Request handshake message. The modified CA field must not be the CA used to sign the client's certificate. The evaluator shall verify that the server rejects the connection after receiving the Client Finished handshake message.

    ○ Modify a byte in the Server Finished handshake message, and verify that the client sends a fatal alert upon receipt and does not send any application data.

5.3 Class: User Data Protection (FDP)

5.3.1 Access Control (FDP_ACF)

FDP_ACF_EXT.1	Extended: Security access control

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:

- get current GPS 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

Assurance Activity:

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.

Assurance 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 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 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 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 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.

5.3.2 Data-At-Rest Protection (FDP_DAR)

FDP_DAR_EXT.1	**Extended: Data-At-Rest Protection**

FDP_DAR_EXT.1.1 Encryption shall cover all protected data.

Application Note: As defined in F.1 Glossary, protected data is all non-TSF data, including all user or enterprise data.

FDP_DAR_EXT.1.2 Encryption shall be performed using DEKs with AES in the [selection: *XTS, CBC, GCM*] mode with key size [selection: *128,256*] bits.

Assurance Activity:

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.

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.

Assurance 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 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**.

5.3.3 Certificate Data Storage (FDP_STG)

FDP_STG_EXT.1(1)	Extended: User Data Storage

FDP_STG_EXT.1.1(1) The TSF shall provide protected storage for the Trust Anchor Database.

Assurance Activity:

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.1**, **FMT_MOF.1(1)**, and **FMT_MOF.1(2)**.

5.4 Class: Identification and Authentication (FIA)

5.4.1 Authentication Failures (FIA_AFL)

FIA_AFL_EXT.1	Authentication failure handling

FIA_AFL_EXT.1.1 The TSF shall detect when [***a configurable positive integer within*** [assignment: *range of acceptable values*]] of unsuccessful authentication attempts occur related to [***last successful authentication by that user***].

Application Note: The positive integer is configured according to **FMT_SMF.1.1** function 2c.

FIA_AFL_EXT.1.2 When the defined number of unsuccessful authentication attempts has been [selection: *met, surpassed*], the TSF shall [***perform*** [selection: *full wipe of all protected data, a remediation action set by the administrator*]

Application Note: Full wipe is performed in accordance with **FCS_CKM_EXT.5**. If the ST Author selects an administrator-configured remediation action in the second selection, the ST Author must list authentication failures as one of the administrator-configured triggers in **FMT_SMF_EXT.1**.

Assurance Activity:

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 user. The evaluator shall verify that the AGD guidance describes how the administrator configures the maximum number of unsuccessful authentication attempts and the remediation action to be performed when that maximum is met or surpassed.

Test 1: The evaluator shall configure according to the AGD guidance the device with a maximum number of unsuccessful authentication attempts and with a remediation action to be performed when that maximum is met or surpassed. The evaluator shall enter the locked state and enter incorrect passwords until the remediation action occurs. The evaluator shall

verify that the number of password entries corresponds to the configured maximum and that the remediation action is implemented.

5.4.2 Port Access Entity Authentication (FIA_PAE)

FIA_PAE_EXT.1	Extended: PAE Authentication

FIA_PAE_EXT.1 The TSF shall conform to [***IEEE Standard 802.1X***] for a Port Access Entity (PAE) in the "Supplicant" role.

Application Note: This requirement covers the TSF's role as the supplicant in an 802.1X authentication exchange. If the exchange is completed successfully, the TSF will derive the PMK as a result of the EAP-TLS (or other appropriate EAP exchange) and perform the 4-way handshake with the wireless access system (authenticator) to begin 802.11 communications.

As indicated previously, there are at least two communication paths present during the exchange; one with the wireless access system and one with the authentication server that uses the wireless access system as a relay. The TSF establishes an EAP over LAN (EAPOL) connection with the wireless access system as specified in 802.1X-2010. The TSF and authentication server establish an EAP-TLS session (RFC 5216).

The point of performing 802.1X authentication is to gain access to the network (assuming the authentication was successful and that all 802.11 negotiations are performed successfully); in the terminology of 802.1X, this means the TSF will gain access to the "controlled port" maintained by the wireless access system.

Assurance Activity:

The evaluator shall perform the following tests:

- *Test 1:* The evaluator shall demonstrate that the TOE has no access to the test network. After successfully authenticating with an authentication server through a wireless access system, the evaluator shall demonstrate that the TOE does have access to the test network.
- *Test 2:* The evaluator shall demonstrate that the TOE has no access to the test network. The evaluator shall attempt to authenticate using an invalid client certificate, such that the EAP-TLS negotiation fails. This should result in the TOE still being unable to access the test network.
- *Test 3:* The evaluator shall demonstrate that the TOE has no access to the test network. The evaluator shall attempt to authenticate using an invalid authentication server certificate, such that the EAP-TLS negotiation fails. This should result in the TOE still being unable to access the test network.

5.4.3 Password Management (FIA_PMG)

FIA_PMG_EXT.1	Extended: 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 TOE; they may optionally list additional special characters supported using the assignment.

Assurance Activity:

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.

Test 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.

5.4.4 Authentication Throttling (FIA_TRT)

FIA_TRT_EXT.1	**Extended: 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*]. The minimum delay shall be such that no more than [***10***] attempts can be attempted per [***500 milliseconds***].

Application Note: The user authentication attempts in this requirement are attempts to guess the Password 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 password brute forcing; for example, the expected time to find a randomly generated password of 4

characters (utilizing the minimum character set of 63 characters) is 4 ½ days, and the time for 5 characters is over 287 days.

Assurance Activity:

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.

5.4.5 User Authentication (FIA_UAU)

5.4.5.1 Protected Authentication Feedback

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: 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.

Assurance Activity:

The evaluator shall ensure that the TSS describes the means of obscuring the password entry. The evaluator shall verify that any configuration of this requirement is addressed in the AGD guidance and that the password is obscured by default.

Test: The evaluator shall enter passwords on the device, including at least the Password Authentication Factor at lockscreen, and verify that the password is not displayed on the device.

5.4.5.2 Authentication for Cryptographic Operation

FIA_UAU_EXT.1	Extended: 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 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 (see F.1 Glossary and D.3.2 Data-At-Rest Protection (FDP_DAR)), which minimally includes software-based secure key storage.

Assurance Activity:

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) DEK(s) for sensitive data, in accordance with **FCS_STG_EXT.2**.

The following tests may be performed in conjunction with **FDP_DAR_EXT.1**.

Assurance 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 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 amongst 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 search for the unique string amongst the application data, and verify that the unique string can be found.

Test 2: [conditional] The evaluator shall require user authentication according to the AGD guidance. The evaluator shall write, or the developer shall provide access to, an application that generates and stores a key in the software-based secure key storage.

The evaluator shall lock the device, use a tool provided by developer to search for the key amongst the application data, and verify that the key cannot be found. The evaluator shall enter the Password Authentication Factor to access full device functionality, use a tool provided by developer to search for the key amongst the application data, and verify that the unique string can be found.

Test 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 (this may be data or a key).

The evaluator shall lock the device, use a tool provided by developer to search for the unique string amongst 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 search for the unique string amongst the application data, and verify that the unique string can be found.

5.4.5.3 Timing of Authentication

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.

Assurance Activity:

The evaluator shall verify that the TSS describes the actions allowed by unauthorized users in the locked state. 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.

5.4.5.4 Re-Authentication

FIA_UAU_EXT.3	Extended: Re-Authentication

FIA_UAU_EXT.3.1: The TSF shall require the user to enter the correct Password Authentication Factor when the user changes the Password Authentication Factor, and following TSF- and user-initiated locking in order to transition to the unlocked state, and [selection:[assignment: *other conditions*], *no other conditions*].

Application Note: TSF- and user-initiated locking is described in **FTA_SSL_EXT.1**.

Assurance Activity:

Test 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 2: The evaluator shall configure the TSF to transition to the locked state after a time of inactivity (**FMT_SMF.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 3: 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.

5.4.6 X509 Certificates (FIA_X509)

5.4.6.1 Validation of Certificates

FIA_X509_EXT.1	Extended: Validation of certificates

FIA_X509_EXT.1.1 The TSF shall validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation.
- The certificate path must terminate with a certificate in the Trust Anchor Database.
- The TSF shall validate a certificate path by ensuring the presence of the basicConstraints extension and that the cA flag is set to TRUE for all CA certificates.
- The TSF shall validate the revocation status of the certificate using [selection: *the Online Certificate Status Protocol (OCSP) as specified in RFC 2560, a Certificate Revocation List (CRL) as specified in RFC 5759*].

- The TSF shall validate the extendedKeyUsage field according to the following rules:
    - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3).
    - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.

Application Note: FIA_X509_EXT.1.1 lists the rules for validating certificates. The ST author shall select whether revocation status is verified using OCSP or CRLs. **FIA_X509_EXT.2** requires that certificates are used for EAP-TLS; this use requires that the extendedKeyUsage rules are verified. Certificates may optionally be used for trusted updates of system software and mobile applications (**FPT_TUD_EXT.2**) and for integrity verification (**FPT_TST_EXT.2**) and, if implemented, must be validated to contain the Code Signing purpose extendedKeyUsage.

While **FIA_X509_EXT.1.1** requires that the TOE perform certain checks on the certificate presented by a TLS server, there are corresponding checks that the authentication server will have to perform on the certificate presented by the client; namely that the extendedKeyUsage field of the client certificate includes "Client Authentication" and that the key agreement bit (for the Diffie-Hellman ciphersuites) or the key encipherment bit (for RSA ciphersuites) be set. Certificates obtained for use by the TOE will have to conform to these requirements in order to be used in the enterprise. This check is required to support EAP-TLS for the WLAN trusted channel.

FIA_X509_EXT.1.2 The TSF shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

Application Note: This requirement applies to certificates that are used and processed by the TSF and restricts the certificates that may be added to the Trust Anchor Database.

Assurance Activity:

The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place. The evaluator ensures the TSS also provides a description of the certificate path validation algorithm.

The tests described must be performed in conjunction with the other Certificate Services assurance activities, including the use cases in **FIA_X509_EXT.2.1** and **FIA_X509_EXT.3**. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules.

Test 1: The evaluator shall demonstrate that validating a certificate without a valid certification path results in the function (application validation, trusted channel setup, or trusted software update) failing. The evaluator shall then load a certificate or certificates 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 function fails.

Test 2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.

Test 3: The evaluator shall test that the TOE can properly handle revoked certificates – conditional on whether CRL or OCSP is selected; if both are selected, and then a test is performed for each method. The evaluator has to only test one up in the trust chain (future revisions may require to ensure the validation is done up the entire chain). The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that will be revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails.

Test 4: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOE's certificate does not contain the basicConstraints extension. The validation of the certificate path fails.

Test 5: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOE's certificate has the cA flag in the basicConstraints extension not set. The validation of the certificate path fails.

Test 6: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOE's certificate has the cA flag in the basicConstraints extension set to TRUE. The validation of the certificate path succeeds.

5.4.6.2 X509 Certificate Authentication

FIA_X509_EXT.2 Extended: X509 certificate authentication

FIA_X509_EXT.2.1 The TSF shall use X.509v3 certificates as defined by RFC 5280 to support authentication for EAP-TLS exchanges, and [selection*: IPsec, TLS, HTTPS, DTLS*]], and [selection: *code signing for system software updates, code signing for mobile applications, code signing for integrity verification, [assignment: other uses], no additional uses*].

Application Note: The ST author's selection shall match the selection of **FTP_ITC_EXT.1.1**. Certificates may optionally be used for trusted updates of system software (**FPT_TUD_EXT.2.3)** and mobile applications (**FPT_TUD_EXT.2.5)** and for integrity verification (**FPT_TST_EXT.2**). If any of the code signing uses is selected, **FIA_X509_EXT.2.4** must be included in the main body. If **FPT_TUD_EXT.2.5** is included in the main body, "code signing for mobile applications" must be included in the selection.

FIA_X509_EXT.2.2 When the TSF cannot establish a connection to determine the validity of a certificate, the TSF shall [selection: *allow the administrator to choose whether to accept the certificate in these cases, accept the certificate, not accept the certificate*].

Application Note: Often a connection must be established to perform a verification of the revocation status of a certificate - either to download a CRL or to perform OCSP. The selection is used to describe the behavior in the event that such a connection cannot be established (for example, due to a network error). If the TOE has determined the certificate valid according to all other rules in **FIA_X509_EXT.1**, the behavior indicated in the second selection shall determine the validity. The TOE must not accept the certificate if it fails any of the other validation rules in **FIA_X509_EXT.1**. If the administrator-configured option is selected by the ST Author, the ST Author must also select function 32 in **FMT_SMF.1**.

FIA_X509_EXT.2.3 The TSF shall not establish a trusted communication channel if the peer certificate is deemed invalid.

Application Note: Trusted communication channels include WLAN and any of IPsec, TLS, HTTPS, or DTLS performed by the TSF. Validity is determined by the certificate path, the expiration date, and the revocation status in accordance with RFC 5280.

Assurance Activity:

The evaluator shall check the TSS to ensure that it describes how the TOE chooses which certificates to use, and any necessary instructions in the administrative guidance for configuring the operating environment so that the TOE can use the certificates.

The evaluator shall examine the TSS to confirm that it describes the behavior of the TOE when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel. If the requirement that the administrator is able to specify the default action, then the evaluator shall ensure that the operational guidance contains instructions on how this configuration action is performed.

The evaluator shall perform Test 1 for each function listed in **FIA_X509_EXT.2.1** that requires the use of certificates:

Test 1: The evaluator shall demonstrate that using a certificate without a valid certification path results in the function failing. Using the administrative guidance, the evaluator shall then load a certificate or certificates 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 function fails.

Test 2: The evaluator shall demonstrate that using a valid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate, and observe that the action selected in **FIA_X509_EXT.2.2** is performed. If the selected action is administrator-configurable, then the evaluator shall follow the operational guidance to determine that all supported administrator-configurable options behave in their documented manner.

5.4.6.3 Request Validation of Certificates

FIA_X509_EXT.3	Extended: Request Validation of certificates

FIA_X509_EXT.3.1 The TSF shall provide a certificate validation service to applications.

FIA_X509_EXT.3.2 The TSF shall respond to the requesting application with the success or failure of the validation.

Assurance Activity:

The evaluator shall verify that the API documentation provided according to Section 6.2.1 includes the security function (certificate validation) described in this requirement. This documentation shall be clear as to which results indicate success and failure.

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**, **FDP_ITC_EXT.1**, **FMT_SMF.1.1** function 14, and **FIA_X509_EXT.1**.

5.5 Class: Security Management (FMT)

Both the user and the administrator may manage the TOE; however, the administrator is not a separate role on the device. 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.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.1(2)**.

Table 12 compares the management functions required by this Protection Profile in the following three requirements (**FMT_MOF.1(1)**, **FMT_MOF.1(2)**, and **FMT_SMF.1**).

5.5.1 Management of Functions in TSF (FMT_MOF)

FMT_MOF.1	**Management of security functions behavior**

FMT_MOF.1.1(1) The TSF shall restrict the ability to [perform] the functions [

1. enroll the TOE in management

[selection:

2. enable/disable the VPN protection,
3. enable/disable [assignment: list of radios],
4. enable/disable data transfer capabilities over [assignment: list of externally accessible hardware ports],
5. enable/disable [assignment: list of protocols where the device acts as a server],
6. enable/disable display notification in the locked state of: [selection:
    a. email notifications,
    b. calendar appointments,
    c. contact associated with phone call notification,
    d. text message notification,
    e. other application,based notification
    f. none]
7. enable/disable developer modes,
8. enable data-at rest protection,
9. enable removable media’s data-at-rest protection,
10. enable/disable local authentication bypass,
11. configure the Access Point Name and proxy used for communications between the cellular network and other networks

12. configure the Bluetooth trusted channel
    a. disable the Discoverable mode
    b. disallow Bluetooth connections using versions 1.0, 1.1, 1.2, 2.0, and [assignment: other Bluetooth version numbers]
    c. [selection: restrict Bluetooth profiles, disable legacy pairing and JustWorks pairing, and [selection: [assignment: other pairing methods], no other pairing methods]],
13. wipe sensitive data
14. import keys/secrets into the secure key storage,
15. destroy user-imported keys/secrets and [selection: no other keys/secrets, [assignment: list of other categories of keys/secrets]] in the secure key storage
16. 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,
17. approve import and removal by applications of X.509v3 certificates in the Trust Anchor Database,
18. configure whether to establish a trusted channel or disallow establishment if the TSF cannot establish a connection to determine the validity of a certificate,
19. enable/disable cellular voice functionality,
20. enable/disable device messaging capabilities,
21. enable/disable the cellular protocols used to connect to cellular network base stations,
22. enable/disable voice command control of device functions,
23. read audit logs kept by the TSF,
24. configure [selection: certificate, public-key] used to validate digital signature on applications,
25. approve exceptions for shared use of keys/secrets by multiple applications
26. approve exceptions for destruction of keys/secrets by applications that did not import the key/secret
27. [assignment: list of other management functions to be provided by the TSF], no other management functions]
28. no other functions]

]] to the user.

Application Notes:

The ST author should select those security management functions which only the user may perform.

For function 1, the enrollment function 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.

The assignment in function number 3 consists of all radios, such as Wi-Fi, GPS, cellular, NFC, and Bluetooth, which can be enabled and disabled.

The assignment in function number 4 consists of all externally accessible hardware ports, such as USB, the SD card, and HDMI, whose data transfer capabilities can be enabled and disabled.

The assignment in function number 5 consists of all protocols where the TSF acts as a server, such as WiFi tethering, which can be enabled and disabled.

The configuration of display of notifications in function number 6 may be those functions which are allowed in the selection of **FTA_SSL_EXT.1.**

Function number 17 may be included in the selection if the TSF allows applications to import or remove X.509v3 certificates from the Trust Anchor Database and only the user may approve requests. These applications do not include MDM Agents. 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.

The ability to perform function number 19 includes the ability to disable voice calls completely (except emergency dialing).

The ability to perform function number 20 includes the ability to disable device messaging completely (except any carrier-required and emergency SMS). Device messaging capabilities include SMS, MMS, and voicemail.

Assurance Activity:

The evaluation shall verify that the TSS describes those management functions which may only be performed by the user in conjunction with the TSS description for **FMT_SMF.1**.

FMT_MOF.1.1(2) The TSF shall restrict the ability to perform the functions [

1. configure password policy:
    a. minimum password length
    b. minimum password complexity
    c. maximum password lifetime
2. configure session locking policy:
    a. screen-lock enabled/disabled
    b. screen lock timeout
    c. number of authentication failures
3. enable/disable [assignment: list of audio or visual collection devices]
4. configure application installation policy by [selection:
    a. specifying authorized application repository(s),
    b. specifying a set of allowed applications and versions (an application whitelist)
    c. denying installation of applications],

[selection:

5. enable/disable the VPN protection
6. enable/disable [assignment: list of radios]
7. enable/disable data transfer capabilities over [assignment: list of externally accessible hardware ports],
8. enable/disable [assignment: list of protocols where the device acts as a server],
9. specify wireless networks (SSIDs) to which the TSF may connect

10. configure security policy for each wireless network:
    a. [selection: specify the CA(s) from which the TSF will accept WLAN authentication server certificate(s), specify the FQDN(s) of acceptable WLAN authentication server certificate(s)]
    b. ability to specify security type
    c. ability to specify authentication protocol
    d. specify the client credentials to be used for authentication
    e. [assignment: any additional WLAN management functions]
11. enable/disable developer modes,
12. enable data-at rest protection,
13. enable removable media's data-at-rest protection,
14. enable/disable local authentication bypass,
15. configure the Access Point Name and proxy used for communications between the cellular network and other networks
16. configure the Bluetooth trusted channel
    a. disable the Discoverable mode
    b. disallow Bluetooth connections using versions 1.0, 1.1, 1.2, 2.0, and [assignment: other Bluetooth version numbers]
    c. [selection: restrict Bluetooth profiles, disable legacy pairing and JustWorks pairing, and [selection: [assignment: other pairing methods], no other pairing methods]],
17. enable/disable display notification in the locked state of: [selection:
    a. email notifications,
    b. calendar appointments,
    c. contact associated with phone call notification,
    d. text message notification,
    e. other application-based notifications,
    f. none]
18. import and remove X.509v3 certificates into/from the Trust Anchor Database,
19. configure whether to establish a trusted channel or disallow establishment if the TSF cannot establish a connection to determine the validity of a certificate,
20. approve import and removal by applications of X.509v3 certificates in the Trust Anchor Database,
21. enable/disable cellular voice functionality,
22. enable/disable device messaging capabilities,
23. enable/disable the cellular protocols used to connect to cellular network base stations,
24. enable/disable voice command control of device functions,
25. configure [selection: certificate, public-key] used to validate digital signature on applications,
26. remove applications
27. update system software
28. install applications
29. approve exceptions for shared use of keys/secrets by multiple applications
30. approve exceptions for destruction of keys/secrets by applications that did not import the key/secret
31. [assignment: list of other management functions to be provided by the TSF], no other management functions]

] to the administrator when the device is enrolled and according to the administrator-configured policy.

Application Notes:

As long as the device is enrolled, 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 additional administrators.

The assignment in function number 3 consists of all audio and visual devices, such as camera and microphone, which can be enabled and disabled.

The assignment in function number 6 consists of all radios, such as Wi-Fi, GPS, cellular, NFC, and Bluetooth, which can be enabled and disabled.

The assignment in function number 7 consists of all externally accessible hardware ports, such as USB, the SD card, and HDMI, whose data transfer capabilities can be enabled and disabled.

The assignment in function number 8 consists of all protocols where the TSF acts as a server, such as WiFi tethering, which can be enabled and disabled.

The security policy in function number 10 addresses security types, such as WPA2-Enterprise, and authentication protocols, such as EAP-TLS.

The configuration of display of notifications in function number 17 may be those functions which are allowed in the selection of **FTA_SSL_EXT.1.**

Function number 20 may be included in the selection if the TSF allows applications to import or remove X.509v3 certificates from the Trust Anchor Database and the administrator may approve requests. These applications do not include MDM Agents. 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 with the ability to globally allow or deny any application requests in order to meet this requirement.

The ability to perform function number 21 includes the ability to disable voice calls completely (except emergency dialing).

The ability to perform function number 22 includes the ability to disable device messaging completely (except any carrier-required and emergency SMS). Device messaging capabilities include SMS, MMS, and voicemail.

Assurance Activity:

Test 1: The evaluator shall use the ***test environment*** to deploy policies to mobile devices.

Test 2: The evaluator shall create policies which collectively include all management functions which are controlled by the (enterprise) administrator and cannot be overridden by the user as defined in **FMT_MOF.1.1(2)**. The evaluator shall apply these policies to devices, attempt to override each setting as the user, and ensure that the TSF does not permit it.

5.5.2 Specification of Management Functions (FMT_SMF)

5.5.2.1 Specification of Management Functions

FMT_SMF.1	Specification of Management Functions

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

1. configure password policy:
    a. minimum password length
    b. minimum password complexity
    c. maximum password lifetime
2. configure session locking policy:
    a. screen-lock enabled/disabled
    b. screen lock timeout
    c. number of authentication failures
3. enable/disable the VPN protection
4. enable/disable [assignment: list of radios]
5. enable/disable [assignment: list of audio or visual collection devices]
6. specify wireless networks (SSIDs) to which the TSF may connect
7. configure security policy for each wireless network:
    a. [selection: specify the CA(s) from which the TSF will accept WLAN authentication server certificate(s), specify the FQDN(s) of acceptable WLAN authentication server certificate(s)]
    b. ability to specify security type
    c. ability to specify authentication protocol
    d. specify the client credentials to be used for authentication
    e. [assignment: any additional WLAN management functions]
8. transition to the locked state
9. full wipe of protected data
10. configure application installation policy by [selection:
    a. specifying authorized application repository(s),
    b. specifying a set of allowed applications and versions (an application whitelist)
    c. denying installation of applications],
11. import keys/secrets into the secure key storage,
12. destroy imported keys/secrets and [selection: no other keys/secrets, [assignment: list of other categories of keys/secrets]] in the secure key storage,
13. import X.509v3 certificates into the Trust Anchor Database,
14. 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,
15. enroll the TOE in management
16. remove applications
17. update system software
18. install applications

[selection:

19. enable/disable data transfer capabilities over [assignment: list of externally accessible hardware ports],
20. enable/disable [assignment: list of protocols where the device acts as a server],
21. enable/disable developer modes,
22. enable data-at rest protection,
23. enable removable media's data-at-rest protection,
24. enable/disable local authentication bypass,
25. configure the Access Point Name and proxy used for communications between the cellular network and other networks
26. configure the Bluetooth trusted channel:
    a. disable the Discoverable mode
    b. disallow Bluetooth connections using versions 1.0, 1.1, 1.2, 2.0, and [assignment: other Bluetooth version numbers]
    c. [selection: restrict Bluetooth profiles, disable legacy pairing and JustWorks pairing, and [selection: [assignment: other pairing methods], no other pairing methods]],
27. enable/disable display notification in the locked state of: [selection:
    a. email notifications,
    b. calendar appointments,
    c. contact associated with phone call notification,
    d. text message notification,
    e. other application-based notifications,
    f. none]
28. wipe sensitive data,
29. alert the administrator,
30. remove Enterprise applications,
31. approve import and removal by applications of X.509v3 certificates in the Trust Anchor Database,
32. configure whether to establish a trusted channel or disallow establishment if the TSF cannot establish a connection to determine the validity of a certificate,
33. enable/disable cellular voice functionality,
34. enable/disable device messaging capabilities,
35. enable/disable the cellular protocols used to connect to cellular network base stations,
36. enable/disable voice command control of device functions,
37. read audit logs kept by the TSF,
38. configure [selection: certificate, public-key] used to validate digital signature on applications,
39. approve exceptions for shared use of keys/secrets by multiple applications,
40. approve exceptions for destruction of keys/secrets by applications that did not import the key/secret,
41. configure the unlock banner,
42. [assignment: list of other management functions to be provided by the TSF], no other management functions]

].

Application Notes:

The assignment in function number 4 consists of all radios, such as Wi-Fi, GPS, cellular, NFC, and Bluetooth, which can be enabled and disabled by either the user or administrator.

The assignment in function number 5 consists of all audio and visual devices, such as camera and microphone, which can be enabled and disabled by either the user or administrator.

The security policy in function number 7 addresses security types, such as WPA2-Enterprise, and authentication protocols, such as EAP-TLS. The CA or FQDN is specified for comparison according to **FCS_TLS_EXT.1.2.**

Full wipe of the TSF is performed according to **FCS_CKM_EXT.5**.

In the future, function 14 may require destruction 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 shall 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 15, the enrollment function 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.

The assignment in function number 19 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.

The assignment in function number 20 consists of all protocols where the TSF acts as a server, such as WiFi tethering, which can be enabled and disabled by either the user or administrator.

Function number 21 must be included in the selection if developer modes are supported by the TSF.

Function number 22 must be included in the selection if data-at-rest protection is not natively enabled.

Function number 23 should be included in the selection if the device supports removable media.

Function number 24 must be included in the selection if local authentication bypass is supported.

If the display of notifications in the locked state is supported, the configuration of these notifications (function 27) must be included in the selection.

Function 28 relates to the optional sensitive data at rest requirements **FDP_DAR_EXT.2** and should be included if **FDP_DAR_EXT.2** is included in the main body of the PP.

Function number 31 must be included in the selection if the TSF allows applications to import or remove X.509v3 certificates from the Trust Anchor Database. These applications do not include MDM Agents. 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 number 32 must be included in the selection if the "administrator-configured option" is selection in **FIA_X509_EXT.2.2**.

The ability to perform function number 33 includes the ability to disable voice calls completely (except emergency dialing).

The ability to perform function number 34 includes the ability to disable device messaging completely (except any carrier-required and emergency SMS). Device messaging capabilities include SMS, MMS, and voicemail.

Function number 38 should be included in the selection if **FPT_TUD_EXT.2.5** is included in the main body and the configurable options is selected.

Function number 39 should be included in the selection if user or administrator is selected in **FCS_STG_EXT.1.4**.

Function number 40 should be included in the selection if user or administrator is selected in **FCS_STG_EXT.1.5**.

Function number 41 must be included in the selection if **FTA_TAB.1** is included in the main body.

Assurance Activity:

The following activities shall take place in the ***test environment*** described in the Assurance Activity for **FPT_TUD_EXT.1.1**, **FPT_TUD_EXT.1.2**, **FPT_TUD_EXT.1.3**, and **PT_TUD_EXT.1.4**. The evaluator shall consult the AGD guidance to perform each of the following tests, iterating each test as necessary if both the user and administrator may perform the function. The following test numbers correspond to the function numbers.

Test 1: The evaluator shall exercise the TSF configuration as the administrator and perform positive and negative tests, with at least two assignments for each variable setting, for each of the following:

- minimum password length
- minimum password complexity
- maximum password lifetime

Test 2: The evaluator shall exercise the TSF configuration as the user and the administrator. The evaluator shall perform positive and negative tests, with at least two assignments for each variable setting, for each of the following.

- screen-lock enabled/disabled
- screen lock timeout
- number of authentication failures (may be combined with test for **FIA_AFL.1**)

Test 3: The evaluator shall exercise the TSF configuration to enable the VPN protection. These configuration actions must be used for the testing of the **FDP_IFC.1.1** requirement.

Test 4: The evaluator shall exercise the TSF configuration as both the user and administrator to enable and disable the state of each radio (e.g. Wi-Fi, GPS, cellular, NFC, Bluetooth) listed by the ST author. For each radio, the evaluator shall use a spectrum analyzer and a RF-shielded environment to verify the existence of signals when the radio is enabled and the absence of signals when the radio is disabled. The evaluator shall verify the absence of signals during device reboot and casual usage.

Test 5: The evaluator shall exercise the TSF configuration as both the user and administrator to enable and disable the state of each audio or visual collection devices (e.g. camera, microphone) listed by the ST author. For each collection device, the evaluator shall disable the device and then attempt to use its functionality.

Test 6: The evaluator shall create a test environment consisting of a wireless access system and an authentication server for the purpose of tests 6 and 7. The evaluator shall specify the wireless network and wireless network settings according to the AGD guidance both as an administrator and as a user. The evaluator shall specify a value for each management function according to the configuration of the test network. Minimally, the evaluator shall test a WPA2 Enterprise network using EAP-TLS. The evaluator shall verify that the TSF can establish a connection to the network.

Test 7: The evaluator shall specify a wireless network with an incorrect value for WLAN authentication server and verify that the mobile device cannot connect to the WLAN. The evaluator shall repeat this test, setting incorrect values for the security type and authentication protocol individually and verify that the mobile device cannot connect to the WLAN.

Test 8 & 9: The evaluator shall use the test environment to instruct the TSF, as the administrator, to command the device to:

- transition to a locked state
- perform a wipe of all data

The evaluator must ensure that the device transitions to the locked state upon command. The evaluator must ensure that this management setup is used when conducting the assurance activities in **FCS_CKM_EXT.5.**

Test 10: The evaluator shall exercise the TSF configuration as the administrator to restrict particular applications, sources of applications, or application installation according to the AGD guidance. The evaluator shall attempt to install denied applications and ensure that this is not possible.

Test 11 & 12: The test of these functions is performed in association with **FCS_STG_EXT.1**.

Test 13: The evaluator shall review the AGD guidance to determine that it describes the steps needed to import, modify, or remove certificates in the Trust Anchor database. The evaluator shall import certificates according to the AGD guidance as the user or as the administrator. The evaluator shall verify that no errors occur during import.

Test 14: The evaluator shall remove an administrator-imported certificate and any other categories of certificates included in the assignment of function 15 from the Trust Anchor Database according to the AGD guidance as the user and as the administrator.

Test 15: The evaluator shall verify that user approval is required to enroll the device into management and includes a description of each type of management function that will be enforced.

Test 16: The evaluator shall attempt to remove applications according to the AGD guidance and verify that the TOE no longer permits users to access those applications or their associated data.

Test 17 & 18: The evaluator shall attempt to update the TSF system software (if updates are available) and install mobile applications and verify that updates correctly install and that the version numbers of the system software and of the mobile applications increase.

Test 19: [conditional] The evaluator shall exercise the TSF configuration to enable and disable data transfer capabilities over each externally accessible hardware ports (e.g. USB, SD card, HDMI) listed by the ST author. The evaluator shall use test equipment for the particular interface to ensure that no low-level signalling is occurring on all pins used for data transfer when they are disabled.

Test 20: [conditional] The evaluator shall attempt to disable each listed protocol in the assignment, which should include tethering uses. The evaluator shall verify that remote devices can no longer access the TOE or TOE resources using any disabled protocols.

Test 21: [conditional] The evaluator shall exercise the TSF configuration as both the user and administrator to enable and disable any developer mode. The evaluator shall test that developer mode access is not available when its configuration is disabled. The evaluator shall verify the developer mode remains disabled during device reboot.

Test 22, 23, & 24: [conditional] The evaluator shall exercise the TSF configuration as both the user and administrator to enable system-wide data-at-rest protection according to the AGD guidance. The evaluator shall ensure that all assurance activities for DAR (see Section 5.3.2) are conducted with the device in this configuration. The evaluator shall disable any "Forgot Password" feature and ensure that the device does not offer any password hints.

Test 25: [conditional]: The evaluator shall establish an APN for the test network, configure the private APN onto the device. The evaluator shall then send packets to the publically routable Internet (perhaps using a tool provided by the developer). The evaluator shall observe that these packets are reaching the APN termination point and not arriving via the carrier's internet access gateway. The evaluator shall repeat the test with a different or invalid APN on the device, and verify that the packets do not reach the APN termination point.

Test 26: [conditional] The evaluator shall disable the Discoverable mode and verify that no new Bluetooth peripherals can connect to the device. The evaluator shall disallow each Bluetooth version and attempt to connect a Bluetooth peripheral to the device. The evaluator shall verify with a Bluetooth protocol analysis tool that the TOE does not perform disabled versions or list the disabled versions as supported by the TOE during pairing negotiations with a Bluetooth peripheral. The evaluator shall, according to the selection, restrict which

pairing mechanisms are allowed by the TOE (via Bluetooth profiles or particular pairing protocols). The evaluator shall verify with a Bluetooth protocol analysis tool that the TOE does not perform disabled pairing mechanism or list the disabled mechanism as supported by the TOE during pairing negotiations with a Bluetooth peripheral.

Test 27: [conditional] For each category of information listed in the AGD guidance, the evaluator shall verify that when that TSF is configured to limit the information according to the AGD, the information is no longer displayed in the locked state.

Test 28: [conditional] The evaluator shall attempt to wipe sensitive data resident on the device according to the administrator guidance. The evaluator shall verify that the data is no longer accessible by the user.

Test 29: [conditional] The evaluator shall configure the device to alert the administrator according to the administrator guidance (for example, by configuring a trigger that causes an alert to the MDM). The evaluator shall verify that the administrator receives an alert for the device.

Test 30: [conditional] The evaluator shall attempt to remove any Enterprise applications from the device by following the administrator guidance. The evaluator shall verify that the TOE no longer permits users to access those applications or their associated data.

Test 31: [conditional] The evaluator shall also verify that the API documentation provided according to Section 6.2.1 includes any security functions (import, modification, or destruction of the Trust Anchor Database) allowed by applications.

If applications may import certificates to the Trust Anchor Database. The evaluator shall write, or the developer shall provide access to, an application that imports a certificate into the Trust Anchor Database. The evaluator shall verify that the TOE requires approval before allowing the application to import the certificate:

- The evaluator shall deny the approvals to verify that the application is not able to import the certificate. Failure of import shall be tested by attempting to validate a certificate that chains to the certificate whose import was attempted (as described in the Assurance Activity for **FIA_X509_EXT.1**).

- The evaluator shall repeat the test, allowing the approval to verify that the application is able to import the certificate and that validation occurs.

If applications may remove certificates in the Trust Anchor Database, the evaluator shall write, or the developer shall provide access to, an application that removes certificates from the Trust Anchor Database. The evaluator shall verify that the TOE requires approval before allowing the application to remove the certificate:

- The evaluator shall deny the approvals to verify that the application is not able to remove the certificate. Failure of removal shall be tested by attempting to validate a certificate that chains to the certificate whose removal was attempted (as described in the Assurance Activity for **FIA_X509_EXT.1**).

The evaluator shall repeat the test, allowing the approval to verify that the application is able to remove/modify the certificate and that validation no longer occurs.

Test 32: [conditional] The test of this function is performed in conjunction with **FIA_X509_EXT.2.2**.

Test 33: [conditional] The evaluator shall attempt to disable all cellular voice functionality according to the administrator guidance. The evaluator shall then attempt to place a call on the TOE as the user and verify that the function fails. The evaluator shall also attempt to call the TOE and verify that the call cannot be completed.

Test 34: [conditional] The evaluator shall attempt to disable all device messaging functionality according to the administrator guidance. The evaluator shall then attempt to send a message on the TOE as the user and verify that the function fails. The evaluator shall also attempt to send a message to the TOE and verify that the message is not received.

Test 35: [conditional] The evaluator shall attempt to disable each cellular protocol according to the administrator guidance. The evaluator shall attempt to connect the device to a cellular network and, using network analysis tools, verify that the device does not allow negotiation of the disabled protocols.

Test 36: [conditional] The evaluator shall attempt to disable voice control functionality and shall verify that the TOE no longer performs any actions upon being given a voice command.

Test 37: [conditional] The evaluator shall attempt to read any device audit logs according to the administrator guidance and verify that the logs may be read. This test may be performed in conjunction with the assurance activity of **FAU_GEN.1**.

Test 38: [conditional] The test of this function is performed in conjunction with **FPT_TUD_EXT.2.5**.

Test 39 & 40: [conditional] The test of these functions is performed in conjunction with **FCS_STG_EXT.1**.

Test 41: [conditional] The test of this function is performed in conjunction with **FTA_TAB.1**.

5.5.2.2 Specification of Remediation Actions

FMT_SMF_EXT.1	**Extended: Specification of Remediation Actions**

FMT_SMF_EXT.1 The TSF shall offer [selection: *transition to the locked state, full wipe of protected data, wipe of sensitive data, alert the administrator, remove Enterprise applications*, [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.

Assurance Activity:

The evaluator shall use the test environment to iteratively configure the device to perform each remediation action in the selection upon unenrollment. The evaluator shall unenroll the device according to AGD guidance and verify that the remediation action configured is performed.

5.6 Class: Protection of the TSF (FPT)

5.6.1 Anti-Exploitation Services (FPT_AEX_EXT)

5.6.1.1 Address-Space Layout Randomization

FPT_AEX_EXT.1	Extended: Anti-Exploitation Services (ASLR)

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 Notes: The 8 unpredictable bits may be provided by the TSF RBG (as specified in **FCS_RBG_EXT.1**) but is not required.

Assurance Activity:

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.

Assurance 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 1: The evaluator shall 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 will 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.

5.6.1.2 Memory Page Permissions

FPT_AEX_EXT.2	Extended: Anti-Exploitation Services (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.

Assurance Activity:

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.

5.6.1.3 Stack Overflow Protection

FPT_AEX_EXT.3	Extended: Anti-Exploitation Services (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.

Assurance Activity:

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.

5.6.1.4 Domain Isolation

FPT_AEX_EXT.4	Extended: 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 that to be protected. In addition to the software, any configuration information that controls or influences the behavior of the TSF software is also to be protected from modification by untrusted applications.

Assurance Activity:

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.

Assurance 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) that make up the TSF. 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 1: The evaluator shall check the "permission settings" for each file in vendor provided list of files that make up the TSF and ensure the settings are appropriate for preventing writing by untrusted applications. The evaluator shall attempt to modify a file of their choosing to ensure the mechanism enforces the permission settings and prevents modification.

Test 2: 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 application's image/data.

5.6.2 Key Storage (FPT_KST)

5.6.2.1 Plaintext Key Storage

FPT_KST_EXT.1	**Extended: 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.

Assurance Activity:

The evaluator shall consult the TSS section of the ST in performing the assurance 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 wrapped with a KEK.

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.

5.6.2.2 No Key Transmission

FPT_KST_EXT.2	Extended: No Key Transmission

FPT_KST_EXT.2.1 The TSF shall not transmit any plaintext key material from the cryptographic module.

Application Note: For the purposes of this requirement, key material refers to keys, passwords, and other material that is used to derive keys. The intention of this requirement is to prevent the logging of plaintext key information to a service that transmits the information off-device.

Assurance Activity:

The evaluator shall consult the TSS section of the ST in performing the assurance activities for this requirement. The evaluator shall ensure that the TSS describes the cryptographic module 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 cryptographic module.

The evaluator shall review the TSS to determine that it makes a case that key material is not transmitted outside the cryptographic module.

5.6.2.3 No Plaintext Key Export

FPT_KST_EXT.3	Extended: No Plaintext Key Export

FPT_KST_EXT.3.1 The TSF shall ensure it is not possible for the TOE user(s) 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.

Assurance Activity:

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.

5.6.3 Self-Test Event Notification (FPT_NOT)

FPT_NOT_EXT.1	Extended: Event 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]].

Assurance Activity:

The evaluator shall verify that the TSS describes critical failures that may occur and the actions to be taken upon these critical failures.

Assurance 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 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.

5.6.4 Reliable Time Stamps (FPT_STM)

FPT_STM.1	Reliable time stamps

FPT_STM.1.1 The TSF shall be able to provide reliable time stamps for its own use.

Assurance Activity:

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 GPS, a NTP server, or the carrier's network time as the primary time sources and whether any or all of these sources is configurable.

The evaluator examines the operational guidance to ensure it instructs the administrator how to set the time. If the TOE supports the use of an NTP server, the operational guidance instructs how a communication path is established between the TOE and the NTP server, and any configuration of the NTP client on the TOE to support this communication.

Test 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.

Test 2: *[conditional]* If the TOE supports the use of an NTP server; the evaluator shall use the operational guidance to configure the NTP client on the TOE, and set up a communication path with the NTP server. The evaluator will observe that the NTP server has set the time to what is expected. If the TOE supports multiple cryptographic protocols for establishing a connection with the NTP server, the evaluator shall perform this test using each supported protocol.

5.6.5 TSF Functionality Testing (FPT_TST)

5.6.5.1 TSF Cryptographic Functionality Testing

FPT_TST_EXT.1 Extended: 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. The self-tests must be performed before the cryptographic functionality is exercised (for example, during the initialization of a process that utilizes the functionality).

Assurance Activity:

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.

5.6.5.2 TSF Integrity Testing

FPT_TST_EXT.2 Extended: TSF Integrity Testing

FPT_TST_EXT.2.1 The TSF shall verify the integrity of the Application Processor bootloader software, Application Processor OS kernel, and [selection: *[assignment: list of other executable code], no other executable code*], stored in mutable media prior to its execution through the use of [selection: *a digital signature using a hardware-protected asymmetric key, a hardware-protected hash*].

Application Note: In order to meet this requirement, the hardware protection may be transitive in nature: a hardware-protected public 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.

The first selection may be used to include additional executable code if **FPT_TST_EXT.2.2** is not included in the main body. At this time, the verification of Baseband Processor software stored in mutable media is not required; however, it may be added in the first assignment. If all executable code (including Baseband Processor software) is verified, **FPT_TST_EXT.2.2** should be included in the main body of the ST.

Assurance Activity:

The evaluator shall verify that the TSS section of the ST includes a description of the boot procedures of the software for the TSF's Application Processor. The evaluator shall ensure that before loading the bootloader for the operating system and the kernel, the bootloader and kernel 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.

5.6.6 Trusted Update (FPT_TUD)

5.6.6.1 Trusted Update: TSF Version Query

FPT_TUD_EXT.1 Extended: Trusted Update: 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.

Assurance Activity:

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 provided guidance documentation.

Test 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.

5.6.6.2 Trusted Update Verification

FPT_TUD_EXT.2 Extended: Trusted Update Verification

FPT_TUD_EXT.2.1 The TSF shall verify [***software updates to the TSF***] using [***a digital signature by the manufacturer***] prior to installing those updates.

Application Note: The digital signature mechanism is implemented in accordance with **FCS_COP.1.1(3)**.

At this time, this requirement does not apply to software updates to the software operating outside the Application Processor. In the future, the Assurance Activity will require testing of the trusted software update mechanism of the software running on other processors of the TSF.

FPT_TUD_EXT.2.2 The boot integrity [selection: ***key, hash***] shall only be updated by [***verified software***].

Application Note: The key or hash updated via this requirement is used verifying software before execution in **FPT_TST_EXT.2**. 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**.

FPT_TUD_EXT.2.3 The digital signature verification key shall [selection: *be validated to a public key in the Trust Anchor Database, match a hardware-protected public key*].

Application Note: The ST author shall indicate the method by which the signing key for system software updates is limited and, if selected in **FPT_TUD_EXT.2.3,** shall 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** and should be selected in **FIA_X509_EXT.2.1**.

Assurance Activity:

The evaluator shall verify that the TSS section of the ST describes the TSF software update mechanism 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 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 the public key 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.

The evaluator shall verify that the ST author has provided evidence that the following tests were performed:

Test 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 2: The tester shall digitally sign the update with a key disallowed by the device and verify that installation fails. The tester shall digitally sign the update with the allowed key and verify that installation succeeds.

Test 3: [conditional] The tester shall digitally sign the update with an invalid certificate and verify that update installation fails. The tester shall digitally sign the application with a certificate that does not have the Code Signing purpose and verify that application installation fails.

FPT_TUD_EXT.2.4 The TSF shall verify [mobile application software] using [***a digital signature mechanism***] prior to installation.

Application Note: This requirement does not necessitate a X.509v3 certificate or certificate validation. X.509v3 certificates and certificate validation are addressed in **FPT_TUD_EXT.2.5**.

Assurance Activity:

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.

Test 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.

5.7 Class: TOE Access (FTA)

5.7.1 Session Locking (FTA_SSL)

5.7.1.1 TSF- and User-initiated locked state

FTA_SSL_EXT.1	Extended: 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 and a user initiated lock, and upon transitioning to the locked state, the TSF shall perform the following operations:

a) clearing or overwriting display devices, obscuring the previous contents;

b) [assignment: *other actions performed upon transitioning to the locked state*].

Application Note: The time interval of inactivity is configured using **FMT_SMF.1** function 2b.

Assurance Activity:

The evaluator shall verify the TSS describes the actions performed upon transitioning to the locked state. 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.

Test 1: The evaluator shall configure the TSF to transition to the locked state after a time of inactivity (**FMT_SMF.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 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**.

5.7.2 Wireless Network Access (FTA_WSE)

FTA_WSE_EXT.1	Extended: Wireless Network Access

FTA_WSE_EXT.1.1 The TSF shall be able to attempt connections to wireless networks specified as acceptable networks as configured by the administrator in **FMT_SMF.1**.

Application Note: The intent of this requirement is to allow the user and administrator to configure the access points to which the TSF may connect and to prevent the TSF from connecting to wireless networks without explicit authorization by the user or the administrator. If, when the device is enrolled, the user is prevented from connecting to wireless networks other than those specified by the enterprise administrator, this management function should be listed in the selection for the requirements **FMT_MOF.1.1(2)**. If the management function is not listed in the selection of **FMT_MOF.1.1(2)** requirements, the

user may perform the management function as an administrator in order to configure and connect to wireless networks.

Assurance Activity:

The assurance activity for this requirement is performed in conjunction with the assurance activity for **FMT_SMF.1**.

5.8 Class: Trusted Path/Channels (FTP)

5.8.1 Trusted Channel Communication (FTP_ITC)

FTP_ITC_EXT.1	**Extended: Trusted channel Communication**

FTP_ITC_EXT.1.1 The TSF shall use 802.11-2012, 802.1X, and EAP-TLS and [selection, *at least one of: IPsec, TLS, DTLS, HTTPS protocol*] to provide a communication channel between itself and another trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from disclosure and detection of 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 a wireless access point.

The ST author shall list which trusted channel protocols are implemented by the Mobile Device. If the ST author selects IPsec, the TSF shall be validated against the "Protection Profile for IPsec Virtual Private Network (VPN) Clients." Annex B contains the requirements for implementing each of the other optional trusted channel protocols. 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.

FTP_ITC_EXT.1.2 The TSF shall permit the TSF and applications to initiate communication via the trusted channel.

Application Note: It is expected that applications will not be able to directly initiate communication via IPsec or WLAN, but may initiate communication within those trusted channels initiated by the TSF. In this respect, applications may not be aware that they are initiating the trusted channel for those protocols.

FTP_ITC_EXT.1.3 The TSF shall initiate communication via the trusted channel for connection to a wireless access point and [assignment: list of functions for which a trusted channel is required].

Assurance Activity:

The evaluator shall verify that the API documentation provided according to Section 6.2.1 includes the security functions (trusted channel) described in these requirements. The evaluator shall write, or the developer shall provide access to, an application that requests trusted channel services by the TSF. The evaluator shall verify that the results from the trusted channel match the expected results according to the API documentation. This application may be used to assist in verifying the trusted channel assurance activities for the protocol requirements.

The evaluator shall examine the TSS to determine that it describes the details of the TOE connecting to an access point in terms of the cryptographic protocols specified in the requirement, along with TOE-specific options or procedures that might not be reflected in the specification. The evaluator shall also confirm that all protocols listed in the TSS are specified and included in the requirements in the ST. The evaluator shall confirm that the operational guidance contains instructions for establishing the connection to the access point. The evaluator shall also perform the following tests:

Test 1: The evaluators shall ensure that the TOE is able to initiate communications with an access point using the protocols specified in the requirement, setting up the connections as described in the operational guidance and ensuring that communication is successful.

Test 2: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data is not sent in plaintext.

6. Security Assurance Requirements

The Security Objectives for the TOE in Section 4 were constructed to address threats identified in Section 3. The Security Functional Requirements (SFRs) in Section 5 are a formal instantiation of the Security Objectives. The PP draws from EAL1 the Security Assurance Requirements (SARs) to frame the extent to which the evaluator assesses the documentation applicable for the evaluation and performs independent testing.

While this section contains the complete set of SARs from the CC, the Assurance Activities to be performed by an evaluator are detailed both in Section 5 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 guides for the TOE. The Assurance Activities listed in the ST (which will be refined by the ITSEF to be TOE-specific, either within the ST or in a separate document) will then be performed by the ITSEF. The ITSEF is also expected to perform all of the actions mandated by the Common Evaluation Methodology (CEM) for EAL1. The results of these activities will be documented and presented (along with the administrative guidance used) for validation.

For each family, "Developer Notes" are provided on the developer action elements to clarify what, if any, additional documentation/activity needs to be provided by the developer. For the content/presentation and evaluator activity elements, additional assurance activities (to those already contained in Section 5 and the CEM for EAL1) are described as a whole for the family, rather than for each element. Additionally, the assurance activities described in this section are complementary to those specified in Section 5.

The TOE security assurance requirements, summarized in Table 1, identify the management and evaluative activities required to meet the objectives identified in Section 4 of this PP.

Assurance Class	**Assurance Components**
Security Target (ASE)	ST introduction (ASE_INT.1)
	Conformance claims (ASE_CCL.1)
	Security objectives for the operational environment (ASE_OBJ.1)
	Extended components definition (ASE_ECD.1)
	Stated security requirements (ASE_REQ.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)
Tests (ATE)	Independent testing – sample (ATE_IND.1)

Assurance Class	Assurance Components
Vulnerability assessment (AVA)	Vulnerability survey (AVA_VAN.1)

Table 1: Security Assurance Requirements

6.1 ASE: Security Target

As per ASE activities defined in CEM.

6.2 ADV: Development

The information about the TOE is contained in the guidance documentation available to the end user as well as the TOE Summary Specification (TSS) portion of the ST. The TOE developer must concur with the description of the product that is contained in the TSS as it relates to the functional requirements. The Assurance Activities contained in Section 5 should provide the ST authors with sufficient information to determine the appropriate content for the TSS section.

6.2.1 Basic Functional Specification (ADV_FSP)

The functional specification describes the Target Security Functions Interfaces (TSFIs). It is not necessary to have a formal or complete specification of these interfaces. Additionally, because TOEs conforming to this PP will necessarily have interfaces to the Operational Environment that are not directly invokable by TOE users, there is little point specifying that such interfaces be described in and of themselves since only indirect testing of such interfaces may be possible. For this PP, the activities for this family should focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation. No additional "functional specification" documentation is necessary to satisfy the assurance activities specified.

The interfaces that need to be evaluated are characterized through the information needed to perform the assurance activities listed, rather than as an independent, abstract list.

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 shall include those interfaces required in this profile.

The API documentation shall clearly indicate to which products and versions each available function applies.

The assurance 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.

Assurance Activity:

There are no specific assurance activities associated with these SARs, except ensuring the information is provided. The functional specification documentation is provided to support the evaluation activities described in Section 5, 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 assurance activities being performed; if the evaluator is unable to perform an activity because the there is insufficient interface information, then an adequate functional specification has not been provided.

6.3 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; and

- instructions to manage the security of the TSF as a product and as a component of the larger operational environment; and
- instructions to provide a protected administrative capability.

Guidance pertaining to particular security functionality must also be provided; requirements on such guidance are contained in the assurance activities specified with each requirement.

6.3.1 Operational User Guidance (AGD_OPE)

Developer action elements:

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

Application Note: The operation 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 assurance 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, administrator (e.g., MDM agent), 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 TOE (including operation following failure or operational error), their consequences, and implications for maintaining secure operation.

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

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

Evaluator action elements:

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

Assurance Activity:

Some of the contents of the operational guidance will be verified by the assurance activities in Section 5 and evaluation of the TOE according to the CEM. 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-parties 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:

1. 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).
2. 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.

6.3.2 Preparative Procedures (AGD_PRE)

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 assurance 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 TOE can be prepared securely for operation.

Assurance Activity:

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.

6.4 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's a reflection on the information to be made available for evaluation at this assurance level.

6.4.1 Labelling of the TOE (ALC_CMC)

This component is targeted at identifying the TOE such that it can be distinguished from other products or versions from the same vendor and can be easily specified when being procured by an end user.

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 labelled with its unique reference.

Evaluator action elements:

ALC_CMC.2.1E The evaluator *shall confirm* that the information provided meets all requirements for content and presentation of evidence.

Assurance Activity:

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.

6.4.2 TOE CM Coverage (ALC_CMS)

Given the scope of the TOE and its associated evaluation evidence requirements, this component's assurance activities are covered by the assurance activities listed for **ALC_CMC.1.**

Developer action elements:

ALC_CMS.2.1D The developer shall provide a configuration list for the TOE.

Content and presentation elements:

ALC_CMS.2.1C The configuration list shall include the following: the TOE itself; and the evaluation evidence required by the SARs.

ALC_CMS.2.2C The configuration list shall uniquely identify the configuration items.

Evaluator action elements:

ALC_CMS.2.1E The evaluator *shall confirm* that the information provided meets all requirements for content and presentation of evidence.

Assurance Activity:

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 assurance 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.

Assurance Activity:

The evaluator shall ensure that the developer has identified (in public-facing development documentation 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 environment(s) are invoked (e.g., compiler 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.

6.4.3 Timely Security Updates (ALC_TSU_EXT)

This component requires the TOE developer, in conjunction with any other necessary parties, to provide information as to how the end-user devices are updated to address security issues in a timely manner. The documentation describes the process of providing updates to the public from the time a security flaw is reported/discovered, to the time an update is released. This description includes the parties involved (e.g., the developer, carriers(s)) and the steps that are performed (e.g., developer testing, carrier testing), including worst case time periods, before an update is made available to the public.

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/firmware.

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).

Evaluator action elements:

ALC_TSU_EXT.2.1E The evaluator *shall confirm* that the information provided meets all requirements for content and presentation of evidence.

6.5 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.

6.5.1 Independent Testing – Conformance (ATE_IND)

Testing is performed to confirm the functionality described in the TSS as well as the administrative (including configuration and operational) documentation provided. The focus of the testing is to confirm that the requirements specified in Section 5 are being met, although some additional testing is specified for SARs in Section 6. The Assurance Activities identify the additional testing activities associated with these components. The evaluator produces a test report documenting the plan for and results of testing, as well as coverage arguments focused on the platform/TOE combinations that are claiming conformance to this PP.

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.

Assurance Activity:

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 CEM and the body of this PP's Assurance Activities. While it is not necessary to have one test case per test listed in

an Assurance 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.

6.6 Class AVA: Vulnerability Assessment

For the first 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.

6.6.1 Vulnerability Survey (AVA_VAN)

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.

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.

Assurance Activity:

As with ATE_IND, 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 determine the vulnerabilities that have been found in network infrastructure 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.

A. Rationale

In this PP, the focus in the initial sections of the document is to use a narrative presentation in an attempt to increase the overall comprehensibility of the threats addressed by Mobile Devices; the methods used to mitigate those threats; and the extent of the mitigation achieved by compliant TOEs. This presentation style does not readily lend itself to a formalized evaluation activity, so this section contains the tabular artefacts that can be used for the evaluation activities associated with this document.

A.1. Security Problem Description

A.1.1. 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.

Assumption Name	Assumption Definition
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 exercises precautions to reduce the risk of loss or theft of the Mobile Device.

Table 2: TOE Assumptions

A.1.2. Threats

The threats listed below are addressed by Mobile Devices and apply to all Mobile Devices.

Threat Name	Threat Definition
T.EAVESDROP	If positioned on a wireless communications channel or elsewhere on the network, attackers may monitor and gain access to data exchanged between the Mobile Device and other endpoints
T.NETWORK	An attacker may initiate communications with the Mobile Device or alter communications between the Mobile Device and other endpoints.
T.PHYSICAL	Loss of confidentiality of user data and credentials may be a result of an attacker gaining physical access to a Mobile Device.
T.FLAWAPP	Malicious or exploitable code could be used knowingly or unknowingly by a developer, possibly resulting in the capability of attacks against the platform's system software.
T.PERSISTENT	An attacker gains and continues to have access the device, resulting it loss of integrity and possible control by both an adversary and legitimate owner.

Table 3: Threats

A.1.3. Organizational Security Policies

No organizational policies have been identified that are specific to Mobile Devices.

A.1.4. Security Problem Definition Correspondence

The following table serves to map the threats and assumptions defined in this PP to the security objectives also defined or identified in this PP.

Threat or Assumption	Security Objectives
A.CONFIG	OE.CONFIG
A.NOTIFY	OE.NOTIFY
A.PRECAUTION	OE.PRECAUTION
T.EAVESDROP	O.COMMS, O.CONFIG, O.AUTH
T.NETWORK	O.COMMS, O.CONFIG, O.AUTH
T.PHYSICAL	O.STORAGE, O.AUTH
T.FLAWAPP	O.COMMS, O.CONFIG, O.AUTH, O.INTEGRITY
T.PERSISTENT	O.INTEGRITY

Table 4: Security Problem Definition Correspondence

A.2. Security Objectives

A.2.1. Security Objectives for the TOE

The following table contains security objectives specific to Mobile Devices.

Security Objective Name	Security Objective Definition
O.COMMS	The TOE will provide the capability to communicate using one (or more) standard protocols as a means to maintain the confidentiality of data that are transmitted outside of the TOE.
O.STORAGE	The TOE will provide the capability to encrypt all user and enterprise data and authentication keys to ensure the confidentiality of data that it stores.
O.CONFIG	The TOE will provide the capability to configure and apply security policies. This ensures the Mobile Device can protect user and enterprise data that it may store or process.
O.AUTH	The TOE will provide the capability to authenticate the user and endpoints of a trusted path to ensure they are communicating with an authorized entity with appropriate privileges.
O.INTEGRITY	The TOE will provide the capability to perform self-tests to ensure the integrity of critical functionality, software/firmware and data has been maintained. The TOE will also provide a means to verify the integrity of downloaded updates.

Table 5: Security Objectives for the TOE

A.2.2. Security Objectives for the Operational Environment

The following table contains security objectives specific to the operational environments for Mobile Devices.

Security Objective Name	Security Objective Definition
OE.CONFIG	TOE administrators will configure the Mobile Device security functions correctly to create the intended security policy
OE.NOTIFY	The Mobile User will immediately notify the administrator if the Mobile Device is lost or stolen.
OE.PRECAUTION	The Mobile User exercises precautions to reduce the risk of loss or theft of the Mobile Device.

Table 6: Security Objectives for the Operational Environment

A.2.3. Security Objective Correspondence

The correspondence between the Security Functional Requirements (SFRs) and Security Objectives identified or defined in this PP is provided in Section 4.

B. Optional Requirements

As indicated in the introduction to this PP, the baseline requirements (those that must be performed by the TOE or its underlying platform) are contained in the body of this PP. Additionally, there are three other types of requirements specified in Appendices B, C, and D.

The first type (in this Appendix) are requirements that can be included in the ST, but do not have to be in order for a TOE to claim conformance to this PP. The second type (in Appendix C) are requirements based on selections in the body of the PP: if certain selections are made, then additional requirements in that appendix will need to be included. The third type (in Appendix D) are components that are not required in order to conform to this PP, but will be included in the baseline requirements in future versions of this PP, so adoption by VPN Client vendors is encouraged. Note that the ST author is responsible for ensuring that requirements that may be associated with those in Appendix B, Appendix C, and/or Appendix D but are not listed (e.g., FMT-type requirements) are also included in the ST.

No optional items have been identified at this time.

C. Selection-Based Requirements

As indicated in the introduction to this PP, the baseline requirements (those that must be performed by the TOE or its underlying platform) are contained in the body of this PP. There are additional requirements based on selections in the body of the PP: if certain selections are made, then additional requirements below will need to be included.

C.1. TLS Protocol (FCS_TLS)

FCS_TLS_EXT.2	TLS Protocol

FCS_TLS_EXT.2.1 The TSF shall implement one or more of the following protocols TLS 1.2 (RFC 5246) and [selection: TLS 1.0 (RFC 2246), TLS 1.1 (RFC 4346)] supporting the following ciphersuites:

Mandatory Ciphersuites:
TLS_RSA_WITH_AES_128_CBC_SHA
Optional Ciphersuites:
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_ SHA256
TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256
TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384

Application Note: The ciphersuites to be tested in the evaluated configuration are limited by this requirement. The ST author should select the optional ciphersuites that are supported; if there are no ciphersuites supported other than the mandatory suites, then "None" should be selected. It is necessary to limit the ciphersuites that can be used in an evaluated configuration administratively on the server in the test environment. The Suite B algorithms listed above (RFC 6460) are the preferred algorithms for implementation.

FCS_TLS_EXT.2.2 The TSF shall not establish a trusted channel if the distinguished name (DN) contained in a certificate does not match the expected DN for the peer.

Application Note: The DN may be in the Subject Name field or the Subject Alternative Name extension of the certificate. The expected DN may either be configured or may be compared to the Domain Name or IP address used by the peer.

Assurance Activity:

The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the ciphersuites supported are specified. The evaluator shall check the TSS to ensure that the ciphersuites specified are identical to those listed for this component. The

evaluator shall also check the operational guidance to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS (for instance, the set of ciphersuites advertised by the TOE may have to be restricted to meet the requirements).

The evaluator shall verify that the TSS describes how the DN in the certificate is compared to the expected DN. If the DN is not compared automatically to the Domain Name or IP address, the evaluator shall ensure that the AGD guidance includes configuration of the expected DN for the connection.

Additional tests may be added in the future to test compliance with RFC 5246. The evaluator shall also perform the following tests:

- *Test 1:* The evaluator shall establish a TLS connection using each of the ciphersuites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an EAP session. It is sufficient to observe the successful negotiation of a ciphersuite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic in an attempt to discern the ciphersuite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).
- *Test 2:* The following test is repeated for each supported certificate signing algorithm supported. The evaluator shall attempt to establish the connection using a server with a server certificate that contains the Server Authentication purpose in the extendedKeyUsage field and verify that a connection is established. The evaluator will then verify that the client rejects an otherwise valid server certificate that lacks the Server Authentication purpose in the extendedKeyUsage field and a connection is not established. Ideally, the two certificates should be identical except for the extendedKeyUsage field.
- *Test 3:* The evaluator shall attempt a connection with a certificate where the DN matches either the configured expected DN or the Domain Name/IP address of the peer. The evaluator shall verify that the TSF is able to successfully connect. The evaluator shall attempt a connection with a certificate where the DN does not match either the configured expected DN or the Domain Name/IP address of the peer. The evaluator shall verify that the TSF is not able to successfully connect.
- *Test 4:* The evaluator shall configure the server to send a certificate in the TLS connection that the does not match the server-selected ciphersuite (for example, send a ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite or send a RSA certificate while using one of the ECDSA ciphersuites.) The evaluator shall verify that the TOE disconnects after receiving the server's Certificate handshake message.
- *Test 5*: The evaluator shall setup a man-in-the-middle tool between the TOE and the server and shall perform the following modifications to the traffic:
    - Modify at least one byte in the server's nonce in the Server Hello handshake message, and verify that the server denies the client's Finished handshake message.
    - Modify the server's selected ciphersuite in the Server Hello handshake message to be a ciphersuite not presented in the Client Hello handshake message. The evaluator shall verify that the client rejects the connection after receiving the Server Hello.

- *(conditional)* If a DHE or ECDHE ciphersuite is supported, modify the signature block in the Server's KeyExchange handshake message, and verify that the client rejects the connection after receiving the Server KeyExchange.
- Modify a byte in a CA field in the Server's Certificate Request handshake message. The modified CA field must not be the CA used to sign the client's certificate. The evaluator shall verify that the server rejects the connection after receiving the Client Finished handshake message.
- Modify a byte in the Server Finished handshake message, and verify that the client sends a fatal alert upon receipt and does not send any application data.

C.2. DTLS Protocol (FCS_DTLS)

FCS_DTLS_EXT.1	DTLS Protocol

FCS_DTLS_EXT.1.1 The TSF shall implement the DTLS protocol in accordance with one or more of [selection: DTLS 1.0 (RFC 4347), DTLS 1.2 (RFC 6347)].

FCS_DTLS_EXT.1.2: The TSF shall implement the requirements in TLS (**FCS_TLS_EXT.2**) for the DTLS implementation, except where variations are allowed according to [selection: DTLS 1.0 (RFC 4347), DTLS 1.2 (RFC 6347.)].

Application Note: Differences between DTLS and TLS are outlined in RFC 4347 and RFC 6347; otherwise the protocols are the same. In particular, for the applicable security characteristics defined for the TSF, the two protocols do not differ. Therefore, all application notes and assurance activities that are listed for TLS apply to the DTLS implementation.

Assurance Activity:

The evaluator shall perform the assurance activities listed for TLS to verify this component

C.3. HTTPS Protocol (FCS_HTTPS)

FCS_HTTPS_EXT.1	HTTPS Protocol

FCS_HTTPS_EXT.1.1 The TSF shall implement the HTTPS protocol that complies with RFC 2818.

Application Note: The ST author must provide enough detail to determine how the implementation is complying with the standard(s) identified; this can be done either by adding elements to this component, or by additional detail in the TSS.

FCS_HTTPS_EXT.1.2 The TSF shall implement HTTPS using TLS (**FCS_TLS_EXT.2**).

Assurance Activity:

The evaluator shall check the TSS to ensure that it is clear on how HTTPS uses TLS to establish an administrative session, focusing on any client authentication required by the TLS protocol vs. security administrator authentication which may be done at a different level of the processing stack. Testing for this activity is done as part of the TLS testing; this may result in additional testing if the TLS tests are done at the TLS protocol level.

D. Objective Requirements

As indicated in the introduction to this PP, the baseline requirements (those that must be performed by the TOE or its underlying platform) are contained in the body of this PP. There are additional requirements that specify security functionality that is desirable and these requirements are contained in this Annex. It is expected that these requirements will transition from objective requirements to baseline requirements in future versions of this PP.

At any time these may be included in the ST such that the TOE is still conformant to this PP.

D.1. Class: Security Audit (FAU)

D.1.1. Audit Data Generation (FAU_GEN)

FAU_GEN.1	Audit Data Generation

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the following auditable events:

1. Start-up and shutdown of the audit functions;
2. All administrative actions;
3. User authentication attempts and success/failure of the attempt;
4. Start-up and shutdown of the OS and kernel
5. Failures of security functions;
6. Integrity verification failures;
7. Software updates;
8. Insertion or removal of removable media;
9. Establishment of a synchronizing connection;
10. Establishment of a trusted channel;
11. [selection: Audit records reaching an administrator-configurable percentage of audit capacity, [assignment: other auditable events derived from this profile]].

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information:

Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event.

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’.

Assurance Activity:

The evaluator shall check the administrative guide and ensure that it lists all of the auditable events and provides a format for audit records. Each audit record 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**.

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.

The evaluator shall test the TOE's ability to correctly generate audit records by having the TOE generate audit records 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 records 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 records generated during testing match the format specified in the administrative guide, and that the fields in each audit record have the proper entries.

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 records are generated as expected.

D.1.2. Security Audit Event Selection (FAU_SEL)

FAU_SEL.1	**Selective Audit**

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; and
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/administrator to invoke. For the ST author, the assignment is used to list any additional criteria or "none".

Assurance Activity:

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 records that are always recorded, regardless of the selection criteria currently being enforced.

The evaluator shall also perform the following tests:

Test 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 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.

D.1.3. Security Audit Storage (FAU_STG)

FAU_STG_EXT.1	Audit Storage Protection

FAU_STG_EXT.1.1 The TSF shall overwrite the oldest stored audit records if the audit trail is full.

Assurance Activity:

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 action(s) taken by the TSF when the audit trail is full. The evaluator shall ensure that the action(s) results in the deletion or overwrite of the oldest stored record.

FAU_STG_EXT.1.2 The TSF shall prohibit unauthorized modification and deletion of the audit trail by unauthorized users.

Assurance Activity:

Test: The evaluator shall attempt to access the audit trail as an unauthorized user and shall verify that the attempt fails.

D.2. Class: Cryptographic Services (FCS)

D.2.1. Random Bit Generation (FCS_RBG)

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

FCS_RBG_EXT.1.4 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 shall not count data input from an application towards the entropy required by **FCS_RBG_EXT.1**. Thus, the TSF shall not allow the only input to the RBG seed to be from an application.

Assurance Activity: The evaluator shall verify that this function is included as an interface to the RBG in the documentation required by Annex E and that the behaviour 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. The evaluator shall also perform the following test.

Test 1: The evaluator shall write, or the developer shall provide, an application that adds entropy to the RBG via the Personalization String. The evaluator shall verify that the request succeeds.

FCS_RBG_EXT.1.5 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.

Assurance Activity: The assurance activity for this requirement is captured in the RBG documentation for Annex E. 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.

D.3. Class: User Data Protection (FDP)

D.3.1. Access Control (FDP_ACF)

FDP_ACF_EXT.1	Extended: Security attribute based access control

FDP_ACF_EXT.1.2 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.

Assurance Activity:

Assurance 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 1: The evaluator shall write, or the developer shall provide, an application which attempts to store a file with both write and execute permissions. The evaluator shall verify that this action fails and that the permissions on the file are not simultaneously write and execute.

Test 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.

FDP_ACF_EXT.1.3 The TSF shall be configurable to enforce an access control policy that prevents [selection: application processes, groups of application processes] from accessing

data stored by other [selection: application processes, groups of application processes]. Exceptions may only be explicitly authorized for such sharing by [selection: the user, the administrator, a common application developer].

Assurance Activity:

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.

Test: The evaluator shall write, or the developer shall provide, two applications, one which saves data containing a unique string and the other which attempts to access that data. The evaluator shall verify that the second application is unable to access the stored unique string. The evaluator shall grant access, either as a user, the administrator, or by using a third application with a common application developer to the first, and verify that the application is able to access the stored unique string.

D.3.2. Data-At-Rest Protection (FDP_DAR)

In the current version of the main body requirements, only two levels of data-at-rest protection are addressed: TSF data and Protected Data (and keys). In the future, an additional level of data-at-rest protection will be added: sensitive data. Table 7 addresses the level of protection required for each level of data-at-rest. If the **FDP_DAR_EXT.2** requirements are not included in the body, all non-TSF data, including data that may otherwise be considered sensitive data is treated at the protected data level. Additional information about these data levels can be found in the glossary (Section F.1).

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

Table 7: Protection of Data Levels

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 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 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.

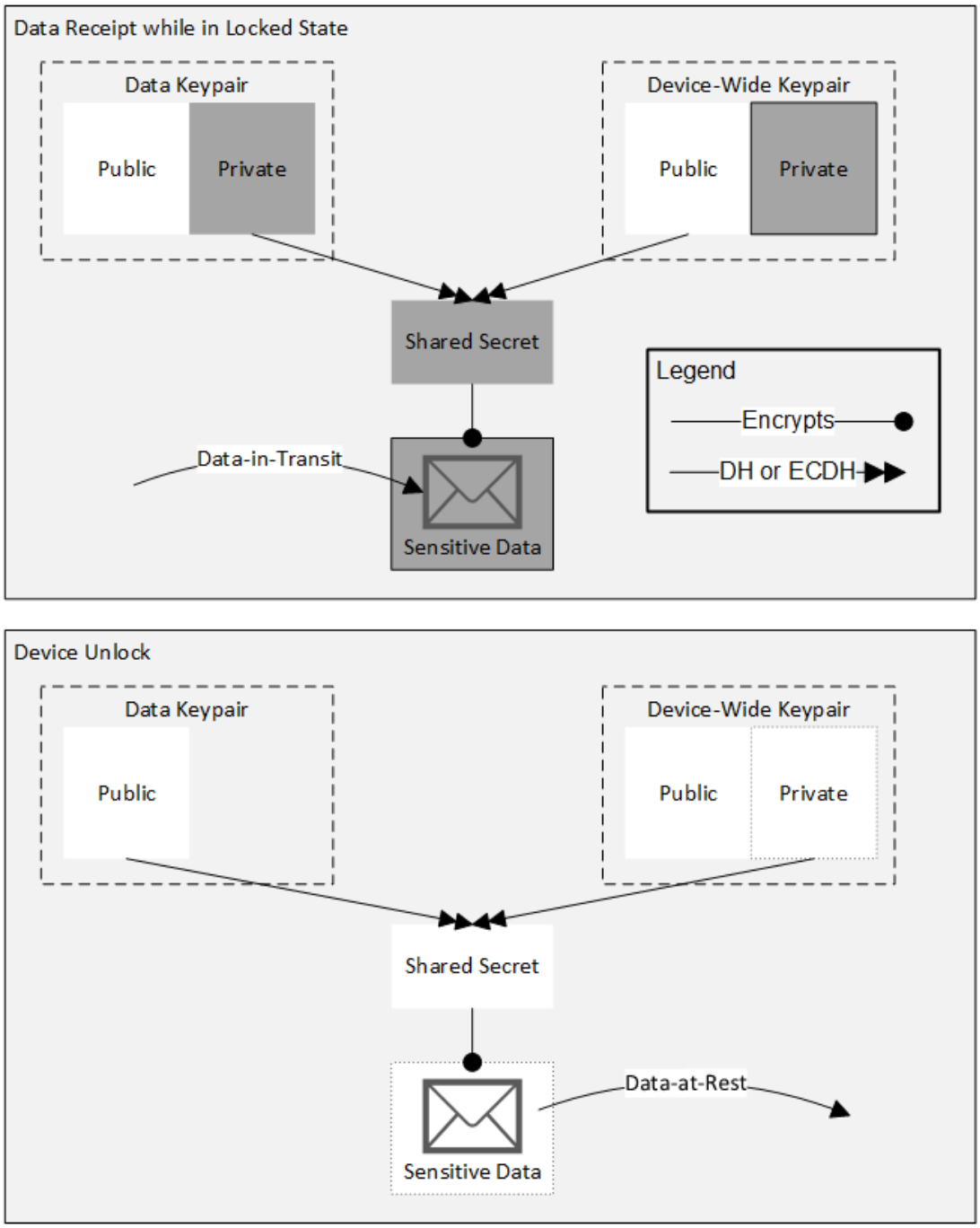

Figure 4: Key Agreement Scheme for Encrypting Received Senstive Data in the Locked State

FDP_DAR_EXT.2	Extended: 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 will 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.

Assurance Activity:

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.).

Test 1: 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 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.1(1)**.

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.4**. 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.

Assurance Activity:

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.

The evaluator shall perform the tests in **FCS_CKM_EXT.4** 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 The TSF shall encrypt any stored symmetric key and any stored private key of the asymmetric key(s) used for the protection of sensitive data according to **FCS_STG_EXT.2** 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 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 KEK.

Assurance Activity:

The evaluator shall verify that the key hierarchy section of the TSS required for **FCS_STG_EXT.2** 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 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 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.

Assurance Activity:

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.

D.3.3. Subset Information Flow Control – VPN (FDP_IFC)

FDP_IFC_EXT.1 Extended: Subset information flow control

FDP_IFC_EXT.1.1 The TSF shall [selection: provide an interface to VPN applications to enable ***all IP traffic (other than IP traffic required to establish the VPN connection) to flow through the IPsec VPN client,*** enable ***all IP traffic (other than IP traffic required to establish the VPN connection) to flow through the IPsec VPN client***].

Application Note: If no native IPsec client is validated or third-party VPN clients may also implement the required Information Flow Control, the second option shall be selected. The ST author shall also identify in the TSS section any differences in the routing of IP traffic when using any supported baseband protocols (e.g. WiFi or, LTE).

The ST author shall select the second option if the TSF implements a native VPN client (IPsec is selected in **FTP_ITC_EXT.1** If this requirement is included in the main body and the native VPN client is to be validated (IPsec is selected in **FTP_ITC_EXT.1** and the TSF is validated against the "Protection Profile for IPsec Virtual Private Network (VPN) Clients"), the ST author shall also include **FDP_IFC_EXT** from the VPN Client Protection Profile.

In the next revision, this requirement will be moved to the main body. In the future, this requirement may also make a distinction between the current requirement (which requires that when the IPsec trusted channel is enabled, all traffic from the TSF is routed through that channel) and having an option to force the establishment of an IPsec trusted channel to allow any communication by the TSF.

Assurance Activity:

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 and 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) is not encapsulated by the VPN protocol (IPsec).

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 IP traffic (other than IP traffic required to establish the VPN connection) through the VPN client.
- The AGD guidance describes how the user and/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.

Test 1: If the ST author identifies any differences in the routing between WiFi 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 WiFi 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 turn on the sniffing tool, establish the VPN connection, and perform the same actions with the device as performed in the first step. 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 IP traffic, aside from and after traffic necessary for establishing the VPN (such as IKE, DNS, and possibly HTTPS), is encapsulated by IPsec. 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 - 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.

D.4. Class: Identification and Authentication (FIA)

D.4.1. Bluetooth Authentication (FIA_BLT)

FIA_BLT_EXT.1	**Extended: Bluetooth Authentication**

FIA_BLT_EXT.1.1 The TSF shall require Bluetooth mutual authentication between devices prior to any data transfer over the Bluetooth link.

Application Note: The mutual authentication is defined by the Bluetooth specification as the pairing mechanism.

Assurance Activity:

The evaluator shall ensure that the TSS describes how data transfer is prevented before the Bluetooth pairing is completed. The TSS shall specifically call out any supported OBEX data transfer mechanisms. The evaluator shall ensure that the OBEX transfers are only completed after the Bluetooth devices are paired.

D.4.2. X509 Certificate Authentication (FIA_X509)

FIA_X509_EXT.2 Extended: X509 certificate authentication

FIA_X509_EXT.2.4 The TSF shall not [selection: *install, execute*] 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.2.5**); in either of these cases, the ST author must select "install". Certificates may optionally be used for code signing for integrity verification (**FPT_TST_EXT.2**); in this case, the ST author must select "execute." If any of the code signing uses is selected in **FIA_X509_EXT.2.1**, **FIA_X509_EXT.2.4** must be included in the main body and the appropriate actions selected.

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

Assurance Activity:

The assurance activity for this requirement is performed in conjunction with the assurance activity for **FIA_X509_EXT.2.1** and **FIA_X509_EXT.2.2**.

FIA_X509_EXT.2.5 The TSF shall generate a Certificate Request Message as specified in RFC 2986 and be able to provide the following information in the request: public key, Common Name, Organization, Organizational Unit, and Country.

Application Note: The public key referenced in **FIA_X509_EXT.2.5** is the public key portion of the public-private key pair generated by the TOE as specified in **FCS_CKM.1(2)**.

Assurance Activity:

The evaluator shall check to ensure that the operational guidance contains instructions on generating a Certificate Request Message.

The evaluator shall also perform the following test.

Test 1: The evaluator shall use the operational guidance to cause the TOE to generate a certificate request message. The evaluator shall confirm that they are able to provide the public key, Common Name, Organization, Organizational Unit, and Country as input into this request. The evaluator shall capture the generated message and ensure that it conforms with the format specified by RFC 2986.

D.5. Class: Security Management (FMT)

D.5.1. Management of Policies (FMT_POL)

FMT_POL_EXT.1 The TSF shall notify the user of any change made by the administrator to the enforced policies.

Application Note: This requirement may be met by providing user access (for example, through menus) to the currently enforced policies rather than perform an alert containing the change. Once the TSF is enrolled, all changes to the policy established at the time of

enrollment require user acceptance. This acceptance may be done on a one-time basis (i.e. "Accept All").

Assurance Activity:

Test: The evaluator shall change an existing policy and deploy it to a device already enrolled. The evaluator shall ensure that the user is notified of the update, if the user has not implicitly permitted the update under the initial enrollment agreement.

D.6. Class: Protection of the TSF (FPT)

D.6.1. Anti-Exploitation Services (FPT_AEX)

FPT_AEX_EXT.1	**Extended: Anti-Exploitation Services (ASLR)**

FPT_AEX_EXT.1.3 The TSF shall provide [***address space layout randomization (ASLR) to the kernel***].

FPT_AEX_EXT.1.4 The base address of any kernel-space memory mapping will consist of at least 4 unpredictable bits.

Application Notes: The 4 unpredictable bits may be provided by the TSF RBG (as specified in **FCS_RBG_EXT.1**).

Assurance Activity:

The evaluator shall ensure that the TSS section of the ST describes how the 4 bits are generated and provides a justification as to why those bits are unpredictable.

Assurance 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 1: The evaluator shall reboot the TOE at least five times. For each of these reboots, the evaluator shall examine memory mapping locations of the kernel. The evaluator must ensure that no memory mappings are placed in the same location on both devices.

FPT_AEX_EXT.2	**Extended: Anti-Exploitation Services (Memory Page Permissions)**

FPT_AEX_EXT.2.2 The TSF shall be able to enforce a policy that write and execute permissions are not simultaneously granted on every page of physical memory.

Application Note: Memory used for just-in-time (JIT) compilation may be excluded from this requirement; if so, the ST author must address how this exception is applied. 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.

Assurance Activity:

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 write XOR execute permissions.

D.6.2. Isolation of Baseband (FPT_BBD)

Mobile devices are becoming increasingly complex having an *application processor* that runs a rich operating system and user applications and separate *baseband processor(s)* 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, WiFi, Bluetooth, FM, GPS) in a single package containing multiple CPUs and DSPs.

Thus, the baseband processor(s) 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.

FPT_BBD_EXT.1 Application Processor Mediation

FPT_BBD_EXT.1.1 Code executing on any baseband processor (BP) shall not be able to access 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)

In future revisions of this Protection Profile, this requirement will be added to the main body.

Assurance Activity:

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.

D.6.3. TSF Integrity Testing (FPT_TST)

FPT_TST_EXT.2 Extended: TSF Integrity Testing

FPT_TST_EXT.2.2 The TSF shall verify the integrity of executable code stored in mutable media when it is loaded for execution through the use of [selection: a digital signature using a hardware-protected asymmetric key, a hardware-protected hash].

Application Note: In order to meet this requirement, the hardware protection may be transitive in nature: a hardware-protected public 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.

Executable code includes the Application Processor bootloader, kernel, OS, device drivers, other processes, applications, and libraries as well as the Baseband Processor bootloader, kernel, and OS.

Because all mutable executable code must be verified, the cryptographic mechanism used to verify the (initial) mutable executable code must be implemented in hardware or in read-only memory (ROM).

Assurance Activity:

The evaluator shall verify that the TSS section of the ST includes a description of the boot procedures of the software for the TSF's application processor and baseband processor. The evaluator shall ensure that before loading any executable code, that code is cryptographically verified. The evaluator shall verify that the TSS contains a justification for the protection of the cryptographic keys or hashes, preventing them from being modified by unverified or unauthenticated software.

D.6.4. Trusted Update (FPT_TUD)

FPT_TUD_EXT.1 Extended: Trusted Update: TSF version query

FPT_TUD_EXT.1.4 The TSF shall cryptographically sign all responses to queries.

Application Note: The queries requiring cryptographically signed responses are **FPT_TUD_EXT.1.1**, **FPT_TUD_EXT.1.2**, and **FPT_TUD_EXT.1.3**. 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.

Assurance Activity:

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. The evaluator shall perform the following test.

Test: The evaluator shall write, or the developer shall provide, a management application that queries each of the responses required in **FPT_TUD_EXT.1**. The evaluator shall verify that the responses to these queries are signed and verify the signatures against the TOE's certificate.

FPT_TUD_EXT.2 Extended: Trusted Update Verification

FPT_TUD_EXT.2.5 The TSF shall by default only accept 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.1** function 38.

Assurance Activity:

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.

Test 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 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 assurance activities for **FIA_X509_EXT.1**.

Test 3: 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.2.6 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.

Assurance Activity:

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.

Test 1: The evaluator shall attempt to install an earlier version of software and shall verify that the update fails.

Test 2: The evaluator shall attempt to install a current or later version and shall verify that the update succeeds.

D.7. Class: TOE Access (FTA)

D.7.1. Default TOE Access Banners (FTA_TAB)

FTA_TAB.1	Default TOE Access Banners

FTA_TAB.1.1 Before establishing a user session, the TSF shall display an Administrator-specified advisory notice and consent warning message regarding 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 shall minimally display this information at startup, but may also display the information at every unlock. The banner is configured according to **FMT_SMF.1** function 41.

Assurance Activity:

The TSS shall describe when the banner is displayed. The evaluator shall also perform the following test:

Test 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.

E. 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.

E.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.

E.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.

E.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.

E.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.

F. Glossary and Acronyms

F.1. Glossary

Term	Meaning
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.
Assurance	Grounds for confidence that a TOE meets the SFRs [CC1].
CC	Common Criteria
CM	Configuration Management
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.
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. Developer modes are states in which additional services are available to a user in order to provide enhanced system access for debugging of software. For the purpose of this profile, these modes also include boot modes which are not verified according to **FPT_TUD_EXT.2**.
Enrolled state	The state in which the Mobile Device is managed with active policy settings from the administrator.
Enterprise Applications	Applications that are provided and managed by the enterprise.
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.
File Encryption Key (FEK)	A DEK used to encrypt a file when File Encryption is used. FEKs are unique to each encrypted file.
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.

Term	Meaning
Locked State	Powered on but most functionality is unavailable for use. User authentication is required to access functionality (when so configured).
MD	Mobile Device
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.
Mobile Device User (User)	This is the person who uses and is held responsible for the mobile device's physical control and operation.
Operating System (OS)	Software which 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 cellular baseband processor and one which runs on the application processor. The OS of the application processor handles most user interaction 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.
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 shutdown.
PP	Protection Profile
Protected Data	Protected data is all non-TSF data, including all user or enterprise data. Protected data is encrypted while the TSF is powered off. Protected data includes all keys in software-based secure key storage. 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.
SAR	Security Assurance Requirement
Sensitive data	Sensitive data shall be identified in the TSS section of the Security Target (ST) by the ST author. 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 optionally protected while in the locked state (**FDP_DAR_EXT.2** and **FDP_DAR_EXT.3**). Sensitive data must minimally include some or all keys in software-based key storage.
SFR	Security Functional Requirement
ST	Security Target
Target of Evaluation	A set of software, firmware and/or hardware possibly accompanied by guidance. [CC1]
TOE	Target of Evaluation

Term	Meaning
TOE Security Functionality (TSF)	A set consisting of all hardware, software, and firmware of the TOE that must be relied upon for the correct enforcement of the SFRs. [CC1]
Trust Anchor Database	A list of trusted root Certificate Authority certificates.
TSF Data	Data for the operation of the TSF upon which the enforcement of the requirements relies.
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).

See [CC1] for other Common Criteria abbreviations and terminology.

F.2. 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
BP	Baseband Processor
CA	Certificate Authority
CBC	Cipher Block Chaining
CCM	Counter with CBC-Message Authentication Code
CCMP	CCM Protocol
CPU	Central Processing Unit
CSP	Cryptographic Service Provider
DAR	Data At Rest
DEK	Data Encryption Key
DEP	Data Execution Prevention
DH	Diffie-Hellman
DN	Distinguished Name
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
FIPS	Federal Information Processing Standards
FM	Frequency Modulation
FQDN	Fully Qualified Domain Name
GCM	Galois Counter Mode
GPS	Global Positioning System
GPU	Graphics Processing Unit
GTK	Group Temporal Key
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

Acronym	Meaning
IPC	Inter-Process Communication
IPsec	Internet Protocol Security
KEK	Key Encryption Key
LTE	Long Term Evolution
MD	Mobile Device
MDM	Mobile Device Management
MMS	Multimedia Messaging Service
NFC	Near Field Communication
NIST	National Institute of Standards and Technology
NX	Never Execute
OID	Object Identifier
OS	Operating System
PAE	Port Access Entity
PBKDF	Password-Based Key Derivation Function
PMK	Pairwise Master Key
PP	Protection Profile
PTK	Pairwise Temporal Key
RBG	Random Bit Generator
REK	Root Encryption Key
ROM	Read-only memory
RSA	Rivest Shamir Adleman
SHA	Secure Hash Algorithm
SMS	Short Messaging Service
SSH	Secure Shell
SSID	Service Set Identifier
ST	Security Target
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Functions
TSS	TOE Summary Specification
USB	Universal Serial Bus
VPN	Virtual Private Network
WiFi	Wireless Fidelity
XCCDF	eXtensible Configuration Checklist Description Format
XTS	XEX (XOR Encrypt XOR) Tweakable Block Cipher with Ciphertext Stealing

G. Use Case Templates

The following use case templates list those selections, assignments, and objective requirements that best support the use cases identified by this Protection Profile. These templates and deviations from the template should be identified in the Security Target to assist customers with making risk-based purchasing decisions. Products that do not meet these templates are not precluded from use in the scenarios identified by this Protection Profile.

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

G.1. [USE CASE 1] Enterprise-owned device for general-purpose enterprise use

Requirement	Action
FAU_GEN.1	Include in ST.
FDP_DAR_EXT.2	Include in ST.
FMT_MOF.1(2) Function 9	Include in selection.
FMT_SMF.1.1 Function 20	Include in selection and assign all protocols where the TSF acts as a server.
FMT_SMF.1.1 Function 25	Include in selection.
FMT_SMF.1.1 Function 37	Include in selection.
FMT_SMF.1.1 Function 41	Include in selection.
FPT_BBD_EXT.1	Include in ST.
FTA_TAB_EXT.1	Include in ST.

Table 8: Enterprise-Owned Template

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

Requirement	Action
FAU_GEN.1	Include in ST.
FCS_CKM.1(1)	Select elliptic curve-based key establishment schemes.
FCS_CKM.1(2)	Select ECDSA schemes.
FCS_CKM_EXT.1	Select 256 bits.
FCS_CKM_EXT.2	Select 256 bits.

FCS_CKM_EXT.3	Select 256 bits.
FCS_COP.1(1)	Select 256 bits.
FCS_COP.1(2)	Select SHA-384.
FCS_COP.1(3)	Select ECDSA schemes.
FCS_COP.1(4)	Select HMAC-SHA-384.
FCS_COP.1(5)	Select HMAC-SHA-384.
FCS_DTLS_EXT.1	Select DTLS 1.2.
FCS_RBG_EXT.1.1	Select a SP800-90A DRBG.
FCS_RBG_EXT.1.2	Select TSF-hardware-based noise source. Select 256 bits.
FCS_STG_EXT.1	Select symmetric keys and persistent secrets. Assurance activities should include ECDSA.
FCS_TLS_EXT.1	Select TLS 1.2. Select TLS_ECDHE_ECDSA_WITH AES_256_GCM_SHA384.
FCS_TLS_EXT.2	Select TLS 1.2. Select TLS_ECDHE_ECDSA_WITH AES_256_GCM_SHA384.
FDP_DAR_EXT.1	Select 256 bits.
FDP_DAR_EXT.2	Include in ST.
FDP_IFC_EXT.1	Include in ST.
FIA_X509.2	Select either “allow the administrator to choose...” or “disallow the establishment of the trusted channel”.
FMT_MOF.1(2) Function 5	Include in selection
FMT_MOF.1(2) Function 9	Include in selection.
FMT_MOF.1(2) Function 10	Include in selection.
FMT_SMF.1.1 Function 4	Assign all radios on TSF.
FMT_SMF.1.1 Function 5	Assign all audio or visual collection devices on TSF.
FMT_SMF.1.1 Function 14	Assign all X.509v3 certificates in the Trust Anchor Database.
FMT_SMF.1.1 Function 19	Include in selection and assign all externally accessible hardware ports.
FMT_SMF.1.1 Function 20	Include in selection and assign all protocols where the TSF acts as a server.
FMT_SMF.1.1 Function 25	Include in selection.
FMT_SMF.1.1 Function 33	Include in selection.
FMT_SMF.1.1 Function 34	Include in selection.

FMT_SMF.1.1 Function 36	Include in selection.
FMT_SMF.1.1 Function 37	Include in selection.
FMT_SMF.1.1 Function 41	Include in selection.
FPT_BBD_EXT.1	Include in ST.
FTA_TAB_EXT.1	Include in ST.

Table 9: High- Security Template

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

Requirement	**Action**
FMT_POL_EXT.1	Include in ST.
FMT_SMF.1.1 Function 12	Include in selection.

Table 10: BYOD Template

H. Initialization Vector Requirements for NIST-Approved Cipher Modes

Cipher Mode	Reference	IV Requirements
Electronic Codebook (ECB)	SP 800-38A	No IV
Counter (CTR)	SP 800-38A	"Initial Counter" shall be non-repeating. No counter value shall be repeated across multiple messages with the same secret key.
Cipher Block Chaining (CBC)	SP 800-38A	IVs shall be unpredictable. Repeating IVs leak information about whether the first one or more blocks are shared between two messages, so IVs should be non-repeating in such situations.
Output Feedback (OFB)	SP 800-38A	IVs shall be non-repeating and shall not be generated by invoking the cipher on another IV.
Cipher Feedback (CFB)	SP 800-38A	IVs should be non-repeating as repeating IVs leak information about the first plaintext block and about common shared prefixes in messages.
XEX (XOR Encrypt XOR) Tweakable Block Cipher with Ciphertext Stealing (XTS)	SP 800-38E	No IV. Tweak values shall be non-negative integers, assigned consecutively, and starting at an arbitrary non-negative integer.
Cipher-based Message Authentication Code (CMAC)	SP 800-38B	No IV
Key Wrap and Key Wrap with Padding	SP 800-38F	No IV
Counter with CBC-Message Authentication Code (CCM)	SP 800-38C	No IV. Nonces shall be non-repeating.
Galois Counter Mode (GCM)	SP 800-38D	IV shall be non-repeating. The number of invocations of GCM shall not exceed 2^32 for a given secret key unless an implementation only uses 96-bit IVs (default length).

Table 11: References and IV Requirements for NIST-approved Cipher Modes

I. Management Functions

Table 12 compares the management functions required by this Protection Profile.

The first column lists the management functions identified in the PP.

The second column indicates whether the function is Mandatory (M) or Optional/Objective (O) to be implemented by the TOE (as listed in **FMT_SMF.1**).

The third column indicates whether the function is a Mandatory (M) or Optional/Objective(O), or Not Applicable (-) function to be restricted to the user (as listed in **FMT_MOF.1(1)**).

The fourth column may be derived from the second and third and indicates whether the function is a Mandatory (M) or Optional/Objective(O), or Not Applicable (-) function to be always available to the administrator. Thus, the TOE must offer these functions, if included in **FMT_SMF.1**, to the administrator to perform.

The fifth column indicates whether the function is a Mandatory (M) or Optional/Objective(O), or Not Applicable (-) function to be restricted to the administrator if the device is enrolled and the administrator applies the indicated policy (as listed in **FMT_MOF.1(2)**).

Management Function **Status Markers:** M – Mandatory O – Optional/Objective - – Not Applicable	**FMT_SMF.1**	**FMT_MOF.1(1)**	**Administrator**	**FMT_MOF.1(2)**
1. configure password policy: a. minimum password length b. minimum password complexity c. maximum password lifetime	M	-	M	M
2. configure session locking policy: a. screen-lock enabled/disabled b. screen lock timeout c. number of authentication failures	M	-	M	M
3. enable/disable the VPN protection	M	O	O	O
4. enable/disable [assignment: list of radios]	M	O	O	O
5. enable/disable [assignment: list of audio or visual collection devices]	M	-	M	M
6. specify wireless networks (SSIDs) to which the TSF may connect	M	-	M	O
7. configure security policy for each wireless network: a. [selection: specify the CA(s) from which the TSF will accept WLAN authentication server certificate(s), specify the FQDN(s) of acceptable WLAN authentication server certificate(s)] b. ability to specify security type c. ability to specify authentication protocol d. specify the client credentials to be used for authentication e. [assignment: any additional WLAN management functions]	M	-	M	O
8. transition to the locked state	M	-	M	-
9. full wipe of protected data	M	-	M	-
10. configure application installation policy by [selection: a. specifying authorized application repository(s),	M	-	M	M

b. specifying a set of allowed applications and versions (an application whitelist), c. denying installation of applications]				
11. import keys/secrets into the secure key storage	M	O	O	-
12. destroy imported keys/secrets and [selection: no other keys/secrets, [assignment: list of other categories of keys/secrets]] in the secure key storage	M	O	O	-
13. import X.509v3 certificates into the Trust Anchor Database	M	-	M	O
14. 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	-
15. enroll the TOE in management	M	M	-	-
16. remove applications	M	-	M	O
17. update system software	M	-	M	O
18. install applications	M	-	M	O
19. enable/disable data transfer capabilities over [assignment: list of externally accessible hardware ports]	O	O	O	O
20. enable/disable [assignment: list of protocols where the device acts as a server]	O	O	O	O
21. enable/disable developer modes	O	O	O	O
22. enable data-at rest protection	O	O	O	O
23. enable removable media's data-at-rest protection	O	O	O	O
24. enable/disable local authentication bypass	O	O	O	O
25. configure the Access Point Name and proxy used for communications between the cellular network and other networks	O	O	O	O
26. configure the Bluetooth trusted channel: a. disable the Discoverable mode b. disallow Bluetooth connections using versions 1.0, 1.1, 1.2, 2.0, and [assignment: other Bluetooth version numbers] c. [selection: restrict Bluetooth profiles, disable legacy pairing and JustWorks pairing, and [selection: [assignment: other pairing methods], no other pairing methods]]	O	O	O	O
27. enable/disable display notification in the locked state of: [selection: a. email notifications, b. calendar appointments, c. contact associated with phone call notification, d. text message notification, e. other application-based notifications, f. none]	O	O	O	O
28. wipe sensitive data	O	O	O	-
29. alert the administrator	O	-	O	-
30. remove Enterprise applications	O	-	O	-
31. approve import and removal by applications of X.509v3 certificates in the Trust Anchor Database	O	O	O	O
32. configure whether to establish a trusted channel or disallow establishment if the TSF cannot establish a connection to determine the validity of a certificate	O	O	O	O
33. enable/disable cellular voice functionality	O	O	O	O
34. enable/disable device messaging capabilities	O	O	O	O
35. enable/disable the cellular protocols used to connect to cellular network base stations	O	O	O	O
36. enable/disable voice command control of device functions	O	O	O	O
37. read audit logs kept by the TSF	O	O	O	
38. configure [selection: certificate, public-key] used to validate digital signature on applications	O	O	O	O
39. approve exceptions for shared use of keys/secrets by multiple applications	O	O	O	O
40. approve exceptions for destruction of keys/secrets by applications that did not import the key/secret	O	O	O	O
41. configure the unlock banner	O	-	O	-

Table 12: Management Functions