Samsung Z with Tizen Version 2.3

Common Criteria Evaluation Security Target

ST Version: 1.0
August 21, 2015

Samsung Electronics Co., Ltd.
416 Maetan-3dong,
Yeongtong-gu, Suwon-si
Gyeonggi-do
443-742
South Korea

Prepared By:

Booz | Allen | Hamilton
delivering results that endure

Cyber Assurance Testing Laboratory
900 Elkridge Landing Road, Suite 100

Linthicum, MD 21090

Table of Contents

1 Security Target Introduction 7
1.1 ST Reference 7
1.1.1 ST Identification 7
1.1.2 Document Organization 7
1.1.3 Terminology 8
1.1.4 Acronyms 10
1.1.5 Reference 11
1.2 TOE Reference 11
1.3 TOE Overview 11
1.4 TOE Type 12
2 TOE Description 13
2.1 Evaluated Components of the TOE 13
2.2 Components and Applications in the Operational Environment 13
2.3 Physical Boundary 13
2.4 Excluded from the TOE 14
2.5 Logical Boundary 14
2.5.1 Cryptographic Support 14
2.5.2 User Data Protection 14
2.5.3 Identification and Authentication 14
2.5.4 Security Management 15
2.5.5 Protection of the TSF 15
2.5.6 TOE Access 15
2.5.7 Trusted Path/Channels 15
3 Conformance Claims 16
3.1 CC Version 16
3.2 CC Part 2 Conformance Claims 16
3.3 CC Part 3 Conformance Claims 16
3.4 PP Claims 16
3.5 Package Claims 16
3.6 Package Name Conformant or Package Name Augmented 16

3.7 Conformance Claim Rationale 16
4 Security Problem Definition 17
4.1 Threats 17
4.2 Assumptions 17
4.3 Security Objectives 17
4.3.1 TOE Security Objectives 18
4.3.2 Security Objectives for the Operational Environment 18
4.4 Security Problem Definition Rationale 18
5 Extended Components Definition 20
5.1 Extended Security Functional Requirements 20
5.2 Extended Security Assurance Requirements 20
6 Security Functional Requirements 21
6.1 Conventions 21
6.2 Security Functional Requirements Summary 21
6.3 Security Functional Requirements 23
6.3.1 Class FCS: Cryptographic Support 23
6.3.2 Class FDP: User Data Protection (FDP) 29
6.3.3 Class FIA: Identification and Authentication 30
6.3.4 Class FMT: Security Management 32
6.3.5 Class FPT: Protection of the TSF 34
6.3.6 Class FTA: TOE Access 37
6.3.7 Class FTP: Trusted Path/Channels 37
6.4 Statement of Security Functional Requirements Consistency 37
7 Security Assurance Requirements 39
7.1 Class ADV: Development 39
7.1.1 Basic Functional Specification (ADV_FSP.1) 39
7.2 Class AGD: Guidance Documentation 40
7.2.1 Operational User Guidance (AGD_OPE.1) 40
7.2.2 Preparative Procedures (AGD_PRE.1) 41
7.3 Class ALC: Life Cycle Support 41
7.3.1 Labeling of the TOE (ALC_CMC.1) 41

7.3.2 TOE CM Coverage (ALC_CMS.1) 42
7.3.3 Timely Security Updates (ALC_TSU_EXT) 42
7.4 Class ASE: Security Target Evaluation 43
7.4.1 Conformance Claims (ASE_CCL.1) 43
7.4.2 Extended Components Definition (ASE_ECD.1) 44
7.4.3 ST Introduction (ASE_INT.1) 45
7.4.4 Security Objectives for the Operational Environment (ASE_OBJ.1) 46
7.4.5 Stated Security Requirements (ASE_REQ.1) 46
7.4.6 TOE Summary Specification (ASE_TSS.1) 47
7.5 Class ATE: Tests 48
7.5.1 Independent Testing - Conformance (ATE_IND.1) 48
7.6 Class AVA: Vulnerability Assessment 48
7.6.1 Vulnerability Survey (AVA_VAN.1) 48
8 TOE Summary Specification 50
8.1 Cryptographic Support 50
8.1.1 FCS_CKM.1(1): 50
8.1.2 FCS_CKM.1(2): 50
8.1.3 FCS_CKM.1(3): 50
8.1.4 FCS_CKM.2: 50
8.1.5 FCS_CKM_EXT.1: 50
8.1.6 FCS_CKM_EXT.2: 51
8.1.7 FCS_CKM_EXT.3: 51
8.1.8 FCS_CKM_EXT.4: 51
8.1.9 FCS_CKM_EXT.5: 51
8.1.10 FCS_CKM_EXT.6: 51
8.1.11 FCS_COP.1(1): 51
8.1.12 FCS_COP.1(2): 52
8.1.13 FCS_COP.1(3): 52
8.1.14 FCS_COP.1(4): 53
8.1.15 FCS_COP.1(5): 53
8.1.16 FCS_HTTPS_EXT.1: 53

8.1.17 FCS_IV_EXT.1: 53
8.1.18 FCS_RBG_EXT.1: 53
8.1.19 FCS_SRV_EXT.1: 53
8.1.20 FCS_STG_EXT.1: 54
8.1.21 FCS_STG_EXT.2: 54
8.1.22 FCS_STG_EXT.3: 55
8.1.23 FCS_TLS_EXT.1: 55
8.1.24 FCS_TLS_EXT.2: 55
8.2 User Data Protection 56
8.2.1 FDP_ACF_EXT.1: 56
8.2.2 FDP_DAR_EXT.1: 56
8.2.3 FDP_STG_EXT.1(1): 56
8.3 Identification and Authentication 57
8.3.1 FIA_AFL_EXT.1: 57
8.3.2 FIA_PAE_EXT.1: 57
8.3.3 FIA_PMG_EXT.1: 57
8.3.4 FIA_TRT_EXT.1: 57
8.3.5 FIA_UAU.7: 57
8.3.6 FIA_UAU_EXT.1: 57
8.3.7 FIA_UAU_EXT.2: 58
8.3.8 FTA_SSL_EXT.1 FIA_UAU_EXT.3: 58
8.3.9 FIA_X509_EXT.1: 58
8.3.10 FIA_X509_EXT.2: 58
8.3.11 FIA_X509_EXT.3: 59
8.4 Security Management 59
8.4.1 FMT_MOF.1(1): 59
8.4.2 FMT_MOF.1(2): 59
8.4.3 FMT_SMF.1: 59
8.4.4 FMT_SMF_EXT.1: 61
8.5 Protection of the TSF 61
8.5.1 FPT_AEX_EXT.1: 61

8.5.2 FPT_AEX_EXT.2: 61
8.5.3 FPT_AEX_EXT.3: 61
8.5.4 FPT_AEX_EXT.4: 61
8.5.5 FPT_KST_EXT.1: 62
8.5.6 FPT_KST_EXT.2: 62
8.5.7 FPT_KST_EXT.3: 62
8.5.8 FPT_NOT_EXT.1: 62
8.5.9 FPT_STM.1: 62
8.5.10 FPT_TST_EXT.1: 63
8.5.11 FPT_TST_EXT.2: 63
8.5.12 FPT_TUD_EXT.1: 63
8.5.13 FPT_TUD_EXT.2: 64
8.6 Timely Security Updates 64
8.6.1 ALC_TSU_EXT.1: 64
8.7 TOE Access 64
8.7.1 FTA_SSL_EXT.1: 64
8.7.2 FTA_WSE_EXT.1: 65
8.8 Trusted Path/Channels 65
8.8.1 FTP_ITC.1: 65

Table of Tables

Table 1-2: CC Specific Terminology 10
Table 1-3: Acronym Definition 11
Table 2-1: Evaluated Components of the TOE 13
Table 2-2: Evaluated Components of the Operational Environment 13
Table 2-3: Operational Environment System Requirements 14
Table 4-1: TOE Threats 17
Table 6-1: Security Functional Requirements for the TOE 23
Table 8-1: OpenSSL Cryptographic Algorithms 52

1 Security Target Introduction

This chapter presents the Security Target (ST) identification information and an overview. An ST contains the Information Technology (IT) security requirements of an identified Target of Evaluation (TOE) and specifies the functional and assurance security measures offered by the TOE.

1.1 ST Reference

This section provides information needed to identify and control this ST and its Target of Evaluation. This ST targets exact compliance with the following Protection Profile (PP):

- Protection Profile for Mobile Device Fundamentals, version 1.1

1.1.1 ST Identification

ST Title:	Samsung Z with Tizen Version 2.3
ST Version:	1.0
ST Publication Date:	August 21, 2015
ST Author:	Booz Allen Hamilton

1.1.2 Document Organization

Chapter 1 of this document provides identifying information for the ST and TOE as well as a brief description of the TOE and its associated TOE type.

Chapter 2 describes the TOE in terms of its physical boundary, logical boundary, exclusions, and dependent Operational Environment components.

Chapter 3 describes the conformance claims made by this ST.

Chapter 4 describes the threats, assumptions, objectives, and organizational security policies that apply to the TOE.

Chapter 5 defines extended Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs).

Chapter 6 describes the SFRs that are to be implemented by the TSF.

Chapter 7 describes the SARs that will be used to evaluate the TOE.

Chapter 8 provides the TOE Summary Specification, which describes how the SFRs that are defined for the TOE are implemented by the TSF.

1.1.3 Terminology

This section defines the terminology used throughout this ST. The terminology used throughout this ST is defined in Table 1-2 and 1-3. These tables are to be used by the reader as a quick reference guide for terminology definitions.

Term	Definition
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) Server Administrator acting through an MDM Agent. If the device is unenrolled, the user is the administrator.
Assurance	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) Server Administrator acting through an MDM Agent. If the device is unenrolled, the user is the administrator.
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**.
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.
Entropy Source	This cryptographic function provides a seed for a random number generator by accumulating the outputs from one or more noise sources. The functionality includes a measure of the minimum work required to guess a given output and tests to ensure that the noise sources are operating properly.
Enrolled state	The state in which the Mobile Device is managed with active policy settings from 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.
FIPS-approved cryptographic function	A security function (e.g., cryptographic algorithm, cryptographic key management technique, or authentication technique) that is either: 1) specified in a Federal Information Processing Standard (FIPS), or 2) adopted in a FIPS and specified either in an appendix to the FIPS or in a document referenced by the FIPS.

Key Chaining	The method of using multiple layers of encryption keys to protect data. A top layer key encrypts a lower layer key which encrypts the data. This method can have any number of layers.
Key Encryption Key (KEK)	A key used to encrypt other keys, such as DEKs or storage that contains keys.
Locked State	Powered on but most functionality is unavailable for use. User authentication is required to access functionality (when so configured).
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.
Security Administrator	Synonymous with Authorized Administrator.
Security Assurance Requirement (SAR)	Description of how assurance is to be gained that the TOE meets the SFR.
Security Functional Requirement (SFR)	Translation of the security objectives for the TOE into a standardized language.
Security Target (ST)	Implementation-dependent statement of security needs for a specific identified TOE.
Target of Evaluation (TOE)	Set of software, firmware and/or hardware possibly accompanied by guidance. For this PP the TOE is Samsung Z with Tizen 2.3.
TOE Security Functionality (TSF)	Combined functionality of all hardware, software, and firmware of a TOE that must be relied upon for the correct enforcement of the SFRs.
TOE Summary	A description of how the TOE satisfies all of the SFRs.

Specification (TSS)	
Trusted Channel	An encrypted connection between the TOE and a trusted remote server.
VPN Gateway	A component that performs encryption and decryption of IP packets as they cross the boundary between a private network and a public network

Table 1-1: CC Specific Terminology

1.1.4 Acronyms

The acronyms used throughout this ST are defined in Table 1-3. This table is to be used by the reader as a quick reference guide for acronym definitions.

Acronym	Definition
AES	Advanced Encryption Standard
ANSI	American National Standards Institute
AP	Application Processor
CA	Certificate Authority
CC	Common Criteria
CLI	Command Line Interface
CMS	Central Management System
CVL	Component Validation List
DEK	Device Encryption Key
DH	Diffie-Hellman
DKEK	Device Key Encryption Key
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
FEK	File Encryption Key
FIPS	Federal Information Processing Standards
FOTA	Firmware Over-the-Air
GUI	Graphical User Interface
HEK	Hardware Encryption Key
HMAC	Hash-based Message Authentication Code
HTTPS	Hypertext Transfer Protocol Secure
IKE	Internet Key Exchange
ISPK	Image Signing Public Key
KCK	Key Confirmation Key
KEK	Key Encryption Key
MD	Mobile Device
MMU	Memory Management Unit
NIST	National Institute of Standards and Technology
ODE	On-Device Encryption
OS	Operating System
PKCS	Public Key Cryptographic Standards
PP	Protection Profile
REK	Root Encryption Key
RGB	Random Bit Generator
RNG	Random Number Generator

SBPK	Secure Boot Public Key
SCP	Secure Copy
SFR	Security Functional Requirement
SHA	Secure Hash Algorithm
SSH	Secure Shell
SSL	Secure Sockets Layer
ST	Security Target
TEE	Trusted Execution Environment
TLS	Transport Layer Security
TSF	TOE Security Functionality
TSS	TOE Summary Specification
TOE	Target of Evaluation
UI	User Interface
VPN	Virtual Public Network

Table 1-2: Acronym Definition

1.1.5 Reference

- Protection Profile for Mobile Device Fundamentals, version 1.1
- Common Criteria for Information Technology Security Evaluation – Part 1: Introduction and general model, dated September 2012, version 3.1, Revision 4, CCMB-2012-009-001
- Common Criteria for Information Technology Security Evaluation – Part 2: Security functional components, dated September 2012, version 3.1, Revision 4, CCMB-2012-009-002
- Common Criteria for Information Technology Security Evaluation – Part 3: Security assurance components, dated September 2012, version 3.1, Revision 4, CCMB-2012-009-003
- Common Methodology for Information Technology Security Evaluation – Evaluation Methodology, dated September 2012, version 3.1, Revision 4, CCMB-2012-009-004

1.2 TOE Reference

The TOE is Samsung Z with Tizen Version 2.3

1.3 TOE Overview

The Target of Evaluation (TOE) is Samsung Z with Tizen version 2.3. The TOE is a mobile device running an operating system based on Linux 3.4 and is being evaluated on the Samsung Z (SM-Z910) hardware platform containing the MSM8974 model processor of the Snapdragon 800 chipset.

The TOE includes a Common Criteria mode (or “CC mode”) that an administrator can invoke through the use of an MDM Server or through a dedicated administrative application. The TOE must be configured as follows in order for an administrator to transition the TOE to CC mode.

- Require a screen lock password (swipe, PIN, pattern, or facial recognition screen locks are not allowed).
- The maximum password failure retry policy should be less than or equal to ten.
- Device encryption must be enabled.

- SDCard encryption must be enabled.
- Revocation checking must be enabled.

When CC mode has been enabled, the TOE behaves as follows.

- The TOE sets the system wide Tizen CC mode property to “Enabled”.
- The TOE performs FIPS 140-2 power-on self-tests.
- The TOE performs self-tests for the key management.
- The TOE performs secure boot integrity checking of the kernel and key system executables.
- The TOE prevents loading of custom firmware/kernels and requires all updates occur through FOTA (Samsung’s Firmware Over The Air firmware update method)
- The TOE uses FIPS 140-2 approved cryptographic ciphers when joining and communicating with wireless networks.
- The TOE utilizes FIPS 140-2 approved cryptographic ciphers for TLS.
- The TOE ensures FOTA updates utilize 2048-bit PKCS #1 RSA-PSS formatted signatures (with SHA-512 hashing).

The TOE combines with a MDM Server solution that enables the enterprise to monitor, control and administer all deployed mobile devices, across multiple mobile service providers as well as facilitate secure communications through a VPN. This partnership provides a secure mobile environment that can be managed and controlled by the environment and reduce the risks that can be introduced through a Bring-Your-Own-Device (BYOD) model.

Data on the TOE is protected through the implementation of Samsung On-Device Encryption (ODE) which utilizes a FIPS 140-2 certified cryptographic module to encrypt device and SD card storage. This functionality is combined with a number of on-device policies including local wipe, remote wipe, password complexity, automatic lock and privileged access to security configurations to prevent unauthorized access to the device and stored data.

The Samsung Tizen Enterprise Software Development Kit (SDK) builds on top of the existing Tizen security model by expanding the security configuration options with additional policies to manage the device.

1.4 TOE Type

The TOE type for Samsung Z with Tizen 2.3 is Mobile Device. The TOE 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. Examples of mobile devices include smartphones, tablet computers, and other mobile devices with similar capabilities.

2 TOE Description

This section provides a description of the TOE in its evaluated configuration. This includes the physical and logical boundaries of the TOE.

2.1 Evaluated Components of the TOE

The following table describes the TOE components in the evaluated configuration:

Component	Definition
Mobile Device	The Samsung Z device and the Tizen version 2.3 software installed on the device.

Table 2-1: Evaluated Components of the TOE

2.2 Components and Applications in the Operational Environment

The following table lists components and applications in the environment that the TOE relies upon in order to function properly:

Component	Required	Definition
Certificate Authority	Yes	A server in the Operational Environment that is responsible for issuing and managing digital certificates.
MDM Server	No	A server in the Operational Environment that is responsible for the administration of Mobile Devices.
Cellular Carrier Time	No	A centralized server provided by the carrier that can be used to provide authoritative system time data to the TOE.
VPN Gateway	Yes	A server in the operational environment that performs encryption and decryption of IP packets as they cross the boundary between a private network and a public network.

Table 2-2: Evaluated Components of the Operational Environment

2.3 Physical Boundary

The physical boundary of the TOE is the Samsung Z (SM-Z910) Mobile Device on which the TOE is installed which includes the MSM8974 model processor of the Snapdragon 800 chipset that has the following specifications:

Component	Details
CPU	Quad-core Krait 400 CPU at up to 2.3 GHz per core
GPU	Qualcomm® Adreno™ 330 GPU
Modem	• Integrated 4G LTE Advanced World Mode, supporting LTE FDD, LTE TDD, WCDMA (DC-HSPA+, DC-HSUPA), CDMA1x, EV-DO Rev. B, TD-SCDMA and GSM/EDGE • 3rd generation integrated LTE modem, with support for LTE-Broadcast
RF	4th generation LTE multimode transceiver with Qualcomm RF360™ Front End solution for world mode bands, lower

	power and PCB reduction
USB	USB 2.0
Bluetooth	BT4.0 integrated digital core
WiFi	1-stream 802.11n/ac Integrated digital core
Memory/Storage	LPDDR3 800MHz Dual-channel 32-bit (12.8GBps)/eMMC 5.0 SATA3 SD 3.0 (UHS-I)

Table 2-3: Operational Environment System Requirements

2.4 Excluded from the TOE

The TOE does not include third-party applications that run on top of the operating system.

2.5 Logical Boundary

The TOE is comprised of several security features. Each of the security features identified above consists of several security functionalities, as identified below.

1. Cryptographic Support
2. User Data Protection
3. Identification and Authentication
4. Security Management
5. Protection of the TSF
6. TOE Access
7. Trusted Path/Channels

2.5.1 Cryptographic Support

The TOE includes a cryptographic module with FIPS 140-2 certified algorithms for a wide range of cryptographic functions including: asymmetric key generation and establishment, symmetric key generation, encryption/decryption, cryptographic hashing and keyed-hash message authentication. These functions are supported with suitable random bit generation, key derivation, salt generation, initialization vector generation, secure key storage, and key and protected data destruction. These primitive cryptographic functions are used to implement security protocols such as TLS and IPsec, and HTTPS and also to encrypt the media (including the generation and protection of data, right, and key encryption keys) used by the TOE. Many if these cryptographic functions are also accessible as services to applications running on the TOE.

2.5.2 User Data Protection

The TOE is designed to control access to system services by hosted applications, including protection of the Trust Anchor Database. Additionally, the TOE is design to protect user and other sensitive data using encryption so that even if a device is physically lost, the data remains protected.

2.5.3 Identification and Authentication

The TOE supports a number of features related to identification and authentication. From a user perspective, except for making phone calls to an emergency number, a password (i.e., Password Authentication Factor) must be correctly entered to unlock the TOE. Also, even when the TOE is unlocked the password must be re-entered to change the password. Passwords are obscured when entered so they cannot be read from the TOE's display. The frequency of entering passwords is limited and when a configured number of failures occur, the TOE will be wiped to protect its contents. Passwords can be constructed using upper and lower cases characters, numbers, and special characters and passwords up to 16 characters are supported.

The TOE can also serve as an 802.1X supplicant and can use X.509v3 and validate certificates for EAP-TLS, TLS and IPsec exchanges.

2.5.4 Security Management

The TOE provides all the interfaces necessary to manage the security functions identified throughout this Security Target as well as other functions commonly found in mobile devices. Several of these functions are accessible by the TOE's users while others are only accessible through the MDM Server and require administrative permissions. Once an enrolled TOE has been un-enrolled, all MDM policies are removed and CC mode is disabled.

2.5.5 Protection of the TSF

The TOE implements a number of features designed to protect itself to ensure the reliability and integrity of its security features. It protects sensitive data such as cryptographic keys so that they are not accessible or exportable. The TOE provides its own timing mechanism to ensure that reliable time information is available (e.g., for log accountability). It enforces read, write, and execute memory page protections, uses address space layout randomization, and stack-based buffer overflow protections to minimize the potential to exploit application flaws. It is also designed to protect itself from modification by applications as well as to isolate the address spaces of applications from one another via sandboxing.

The TOE includes functions to perform self-tests and software/firmware integrity checking so that it might detect when it is failing or may be corrupt. If any of the self-tests fail, the TOE will not go into an operational mode. It also includes mechanisms (i.e., verification of the digital signature of each new image) so that the TOE itself can be updated while ensuring that the updates will not introduce malicious software or other unexpected changes in the TOE. Digital signature checking also extends to verifying applications prior to their installation.

2.5.6 TOE Access

Locking the TOE will obscure its display, which can be manually done by the user or automatically by the TOE after a configured interval of inactivity. The TOE also has the capability to display an advisory message (banner) when a user unlocks the TOE.

The TOE allows an administrator to specify (via the MDM Server) wireless networks to which the user can have the TOE connect.

2.5.7 Trusted Path/Channels

The TOE supports the use of 802.11-2012, 802.1X, EAP-TLS, TLS and IPsec to secure communications channels between itself and other trusted network devices.

3 Conformance Claims

3.1 CC Version

This ST is compliant with Common Criteria for Information Technology Security Evaluation, Version 3.1 Revision 4 September 2012.

3.2 CC Part 2 Conformance Claims

This ST and Target of Evaluation (TOE) is Part 2 extended to include all applicable NIAP and International interpretations through 21 August 2015.

3.3 CC Part 3 Conformance Claims

This ST and Target of Evaluation (TOE) is Part 3 extended to include all applicable NIAP and International interpretations through 21 August 2015.

3.4 PP Claims

This ST claims exact conformance to the following Protection Profiles:

- Protection Profile for Mobile Device Fundamentals, version 1.1

3.5 Package Claims

The TOE claims exact conformance to the above Protection Profile, which extends CC Part 3.

3.6 Package Name Conformant or Package Name Augmented

This ST claims exact conformance with a Protection Profile. The ST is conformant to the claimed package.

3.7 Conformance Claim Rationale

The PP states the following: “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…. 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 TOE type for Samsung Z with Tizen 2.3 is Mobile Device. Therefore, the conformance claim is appropriate.

4 Security Problem Definition

4.1 Threats

This section identifies the threats against the TOE. These threats have been taken from the PP.

Threat	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 4-1: TOE Threats

4.2 Assumptions

These assumptions are made on the operational environment in order to be able to ensure that the security functionality specified in the PP can be provided by the TOE. If the TOE is placed in an operational environment that does not meet these assumptions, the TOE may no longer be able to provide all of its security functionality.

Threat	Threat 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 user exercises precautions to reduce the risk of loss or theft of the Mobile Device.

Table 4-2: TOE Assumptions

4.3 Security Objectives

This section identifies the security objectives of the TOE and its supporting environment. The security objectives identify the responsibilities of the TOE and its environment in meeting the security needs.

4.3.1 TOE Security Objectives

This section identifies the security objectives of the TOE. These objectives have been taken from the Protection Profile for Mobile Device Fundamentals, Version 1.1. The following table contains the objectives defined in the PP:

Objective	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 4-3: TOE Objectives

4.3.2 Security Objectives for the Operational Environment

The Operational Environment of the TOE implements technical and procedural measures to assist the TOE in correctly providing its security functionality (which is defined by the security objectives for the TOE). This section defines the security objectives that are to be addressed by the IT domain or by non-technical or procedural means.

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 4-4: TOE Operational Environment Objectives

4.4 Security Problem Definition Rationale

The assumptions, threats, OSPs, and objectives that are defined in this ST represent the assumptions, threats, OSPs, and objectives that are specified in the Protection Profile to which the TOE claims conformance. The associated mappings of assumptions to environmental objectives, SFRs to TOE objectives, and OSPs and objectives to threats are therefore identical to the mappings that are specified in the claimed Protection Profile.

5 Extended Components Definition

5.1 Extended Security Functional Requirements

The extended Security Functional Requirements that are claimed in this ST are taken directly from the PP to which the ST and TOE claim conformance. These extended components are formally defined in the PP in which their usage is required.

5.2 Extended Security Assurance Requirements

There are no extended Security Assurance Requirements in this ST.

6 Security Functional Requirements

6.1 Conventions

The CC permits four functional component operations—assignment, refinement, selection, and iteration—to be performed on functional requirements. This ST will highlight the operations in the following manner:

- **Assignment:** allows the specification of an identified parameter. Indicated with bold and italicized text.
- **Refinement:** allows the addition of details. Indicated with italicized text.
- **Selection:** allows the specification of one or more elements from a list. Indicated with underlined text.
- **Iteration:** allows a component to be used more than once with varying operations. Indicated with a sequential number in parentheses following the element number of the iterated SFR.

When multiple operations are combined, such as an assignment that is provided as an option within a selection or refinement, a combination of the text formatting is used.

If SFR text is reproduced verbatim from text that was formatted in a claimed PP (such as if the PP's instantiation of the SFR has a refinement or a completed assignment), the formatting is not preserved. This is so that the reader can identify the operations that are performed by the ST author as opposed to the PP author.

6.2 Security Functional Requirements Summary

The following table lists the SFRs claimed by the TOE:

Class Name	Component Identification	Component Name
Cryptographic Support	FCS_CKM.1(1)	Cryptographic Key Generation
	FCS_CKM.1(2)	Cryptographic Key Generation
	FCS_CKM.1(3)	Cryptographic Key Generation
	FCS_CKM.2	Cryptographic Key Distribution
	FCS_CKM_EXT.1	Cryptographic Key Support
	FCS_CKM_EXT.2	Cryptographic Key Random Generation
	FCS_CKM_EXT.3	Cryptographic Key Generation
	FCS_CKM_EXT.4	Key Destruction
	FCS_CKM_EXT.5	TSF Wipe
	FCS_CKM_EXT.6	Salt Generation
	FCS_COP.1(1)	Cryptographic operation
	FCS_COP.1(2)	Cryptographic operation
	FCS_COP.1(3)	Cryptographic operation
	FCS_COP.1(4)	Cryptographic operation
	FCS_COP.1(5)	Cryptographic operation
	FCS_HTTPS_EXT.1	HTTPS Protocol

Class Name	Component Identification	Component Name
	FCS_IV_EXT.1	Initialization Vector Generation
	FCS_RBG_EXT.1	Cryptographic Operation (Random Bit Generation)
	FCS_SRV_EXT.1	Cryptographic Algorithm Services
	FCS_STG_EXT.1	Cryptographic Key Storage
	FCS_STG_EXT.2	Encrypted Cryptographic Key Storage
	FCS_STG_EXT.3	Integrity of encrypted key storage
	FCS_TLS_EXT.1	EAP TLS Protocol
User Data Protection	FDP_ACF_EXT.1	Security access control
	FDP_DAR_EXT.1	Data-At-Rest Protection
	FDP_STG_EXT.1(1)	User Data Storage
Identification and Authentication	FIA_AFL_EXT.1	Authentication failure handling
	FIA_PAE_EXT.1	PAE Authentication
	FIA_PMG_EXT.1	Password Management
	FIA_TRT_EXT.1	Authentication Throttling
	FIA_UAU.7	Protected Authentication Feedback
	FIA_UAU_EXT.1	Authentication for Cryptographic Operation
	FIA_UAU_EXT.2	Timing of Authentication
	FIA_UAU_EXT.3	Re-Authentication
	FIA_X509_EXT.1	Validation of certificates
	FIA_X509_EXT.2	X509 certificate authentication
	FIA_X509_EXT.3	Request Validation of certificates
Security Management	FMT_MOF.1	Management of Security Functions Behavior
	FMT_SMF.1	Specification of Management Functions
	FMT_SMF_EXT.1	Specification of Remediation Actions
Protection of the TSF	FPT_AEX_EXT.1	Anti-Exploitation Services (ASLR)
	FPT_AEX_EXT.2	Anti-Exploitation Services (Memory Page Permissions)
	FPT_AEX_EXT.3	Anti-Exploitation Services (Stack Overflow Protection)
	FPT_AEX_EXT.4	Domain Isolation
	FPT_KST_EXT.1	Key Storage
	FPT_KST_EXT.2	No Key Transmission
	FPT_KST_EXT.3	No Plaintext Key Export
	FPT_NOT_EXT.1	Event Notification
	FPT_STM.1	Reliable time stamps
	FPT_TST_EXT.1	TSF Cryptographic Functionality Testing
	FPT_TST_EXT.1	TSF Integrity Testing
	FPT_TUD_EXT.1	Trusted Update: TSF version query
	FPT_TUD_EXT.2	Trusted Update Verification

Class Name	Component Identification	Component Name
TOE Access	FTA_SSL_EXT.1	TSF- and User-initiated locked state
	FTA_WSE_EXT.1	Wireless Network Access
Trusted Path /Channels	FTP_ITC_EXT.1	Trusted channel Communication

Table 6-1: Security Functional Requirements for the TOE

6.3 Security Functional Requirements

6.3.1 Class FCS: Cryptographic Support

6.3.1.1 FCS_CKM.1(1) 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
- [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 [P-521] (as defined in FIPS PUB 186-4, "Digital Signature Standard")]

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

6.3.1.2 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 [

- FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4 for ECDSA schemes and implementing "NIST curves" P-256, P-384 and [P-521];
- 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.

6.3.1.3 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.

6.3.1.4 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.

6.3.1.5 FCS_CKM_EXT.1 Cryptographic Key Support

FCS_CKM_EXT.1.1

The TSF shall support a hardware-protected REK with an AES key of size [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.

6.3.1.6 FCS_CKM_EXT.2 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 [128, 256] bits.

6.3.1.7 FCS_CKM_EXT.3 Cryptographic Key Generation

FCS_CKM_EXT.3.1

All KEKs shall be [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 [

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 [encrypting one key with another]].

6.3.1.8 FCS_CKM_EXT.4 Key Destruction

FCS_CKM_EXT.4.1

The TSF shall destroy cryptographic keys in accordance with the specified cryptographic key destruction method [

- by clearing the KEK encrypting the target key,
- in accordance with the following rules:
  - For volatile flash memory the destruction shall be executed by [a single direct overwrite consisting of zeroes followed by a read-verify]]

FCS_CKM_EXT.4.2

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

6.3.1.9 FCS_CKM_EXT.5 TSF Wipe

FCS_CKM_EXT.5.1

The TSF shall wipe all protected data by [

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

FCS_CKM_EXT.5.2

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

6.3.1.10 FCS_CKM_EXT.6 Salt Generation

FCS_CKM_EXT.6.1

The TSF shall generate all salts using a RBG that meets FCS_RBG_EXT.1

6.3.1.11 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
- [AES Key Wrap (KW) (as defined in NIST SP 800-38F), AES-GCM (as defined in NIST SP 800-38D)]

and cryptographic key sizes 128-bit key sizes and [256-bit key sizes].

6.3.1.12 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 [SHA-256, SHA-384, SHA-512] and message digest sizes 160 and [256, 384, 512 bits] that meet the following: FIPS Pub 180-4.

6.3.1.13 FCS_COP.1(3) 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)‖, Section 4 for RSA schemes [FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5 for ECDSA schemes and implementing "NIST curves" P-256, P-384 and [P-521]].

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

6.3.1.14 FCS_COP.1(4) 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 [HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512] and cryptographic key sizes [***160, 256, 384, and 512 bits***] and message digest sizes 160 and [256, 384, 512] bits that meet the following: FIPS Pub 198-1, "The Keyed-Hash Message Authentication Code, and FIPS Pub 180-4, "Secure Hash Standard.

6.3.1.15 FCS_COP.1(5) Cryptographic operation

FCS_COP.1.1(5)

The TSF shall perform [Password-based Key Derivation Functions] in accordance with a specified cryptographic algorithm HMAC-[SHA-512] and output cryptographic key sizes [256] that meet the following: NIST SP 800-132.

6.3.1.16 FCS_HTTPS_EXT.1 HTTPS Protocol

FCS_HTTPS_EXT.1.1

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

FCS_HTTPS_EXT.1.2

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

6.3.1.17 FCS_IV_EXT.1 Initialization Vector Generation

FCS_IV_EXT.1.1

The TSF shall generate IVs in accordance with • Protection Profile for Mobile Device Fundamentals, version 1.1 Table 11: References and IV Requirements for NIST-approved Cipher Modes.

6.3.1.18 FCS_RBG_EXT.1 Cryptographic Operation (Random Bit Generation)

FCS_RBG_EXT.1.1

The TSF shall perform all deterministic random bit generation services in accordance with [NIST Special Publication 800-90A using [Hash_DRBG (any), HMAC_DRBG (any), CTR_DRBG (AES)]; 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 [TSF-hardware-based noise source] with a minimum of [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.

6.3.1.19 FCS_SRV_EXT.1 Cryptographic Algorithm Services

FCS_SRV_EXT.1.1

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

- FCS_COP.1(1)
- FCS_COP.1(3)
- FCS_COP.1(2)
- FCS_COP.1(4)
- FCS_COP.1(5)
- FCS_CKM.1(1)
- [FCS_CKM.1(2)]

6.3.1.20 FCS_STG_EXT.1 Cryptographic Key Storage

FCS_STG_EXT.1.1

The TSF shall provide secure key storage for asymmetric private keys and [no other keys].

FCS_STG_EXT.1.2

The TSF shall be capable of importing keys/secrets into the secure key storage upon request of [the user] and [applications running on the TSF].

FCS_STG_EXT.1.3

The TSF shall be capable of destroying keys/secrets in the secure key storage upon request of [the user].

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 [a common application developer].

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 [a common application developer].

6.3.1.21 FCS_STG_EXT.2 Encrypted Cryptographic Key Storage

FCS_STG_EXT.2.1

The TSF shall encrypt all DEKs and KEKs and [all software-based key storage] by KEKs that are [

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

FCS_STG_EXT.2.2

All keys shall be encrypted using AES in the [GCM, CBC mode].

6.3.1.22 FCS_STG_EXT.3 Integrity of encrypted key storage

FCS_STG_EXT.3.1

The TSF shall protect the integrity of any encrypted KEK by [[GCM] cipher mode for encryption according to FCS_STG_EXT.2;]

FCS_STG_EXT.3.2

The TSF shall verify the integrity of the [hash] of the stored key prior to use of the key.

6.3.1.23 FCS_TLS_EXT.1 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 [no other TLS versions] supporting the following ciphersuites:

- Mandatory Ciphersuites in accordance with RFC 3268:

    - 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]

FCS_TLS_EXT.1.2

The TSF shall verify that the server certificate presented for EAP-TLS [chains to one of the specified CAs].

6.3.1.24 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 [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 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]]

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.

6.3.2 Class FDP: User Data Protection (FDP)

6.3.2.1 FDP_ACF_EXT.1 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.

6.3.2.2 FDP_DAR_EXT.1 Data-At-Rest Protection

FDP_DAR_EXT.1.1

Encryption shall cover all protected data.

FDP_DAR_EXT.1.2

Encryption shall be performed using DEKs with AES in the [<u>CBC, GCM</u>] mode with key size [<u>128, 256</u>] bits.

6.3.2.3 FDP_STG_EXT.1(1)User Data Storage

FDP_STG_EXT.1.1(1)

The TSF shall provide protected storage for the Trust Anchor Database.

6.3.3 Class FIA: Identification and Authentication

6.3.3.1 FIA_AFL_EXT.1 Authentication failure handling

FIA_AFL_EXT.1.1

The TSF shall detect when [a configurable positive integer within [***1-99***] of unsuccessful authentication attempts occur related to [last successful authentication by that user].

FIA_AFL_EXT.1.2

When the defined number of unsuccessful authentication attempts has been [<u>met</u>], the TSF shall [perform [<u>full wipe of all protected data</u>]]

6.3.3.2 FIA_PAE_EXT.1 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.

6.3.3.3 FIA_PMG_EXT.1 Password Management

FIA_PMG_EXT.1.1

The TSF shall support the following for the Password Authentication Factor:

1. Passwords shall be able to be composed of any combination of [*upper and lower case letters,* numbers, and special characters: [<u>*“!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”*</u>];
2. Password length up to [***16***] characters shall be supported.

6.3.3.4 FIA_TRT_EXT.1 Authentication Throttling

FIA_TRT_EXT.1.1

The TSF shall limit automated user authentication attempts by [<u>preventing authentication via an external port, enforcing a delay between incorrect authentication attempts</u>]. The minimum delay shall be such that no more than 10 attempts can be attempted per 500 milliseconds.

6.3.3.5 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.

6.3.3.6 FIA_UAU_EXT.1 Authentication for Cryptographic Operation

FIA_UAU_EXT.1.1

The TSF shall require the user to present the Password Authentication Factor prior to decryption of protected data and keys at startup.

6.3.3.7 FIA_UAU_EXT.2 Timing of Authentication

FIA_UAU_EXT.2.1

The TSF shall allow [***[make emergency calls, take screenshots, receive calls, enable/disable airplane mode, turn off, restart, enable Bluetooth, place an outgoing phone call to the last incoming call immediately after the las incoming call is disconnected]***] 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.

6.3.3.8 FIA_UAU_EXT.3 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 [***no other conditions***].

6.3.3.9 FIA_X509_EXT.1 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 [the Online Certificate Status Protocol (OCSP) as specified in RFC 2560].
- The TSF shall validate the extendedKeyUsage field according to the following rules:

- Certificates used for trusted updates and execu 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.

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.

6.3.3.10 FIA_X509_EXT.2 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 [<u>IPsec</u>], and [<u>no additional uses</u>].

FIA_X509_EXT.2.2

When the TSF cannot establish a connection to determine the validity of a certificate, the TSF shall [<u>not accept the certificate</u>].

FIA_X509_EXT.2.3

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

6.3.3.11 FIA_X509_EXT.3 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.

6.3.4 Class FMT: Security Management

6.3.4.1 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

[2. <u>enable/disable the VPN protection</u>

<u>5. enable/disable [***personal hotspot connections and tethered connections***]</u>]

] to the user.

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 [***camera, microphone***]
4. configure application installation policy by [<u>c. denying installation of applications</u>],

<u>[6. enable/disable [***Bluetooth, WiFi***]</u>

<u>31. [***enable and disable location services***]</u>]

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

6.3.4.2 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 [***Bluetooth, WiFi***]

5. enable/disable [***camera, microphone***]

6. specify wireless networks (SSIDs) to which the TSF may connect

7. configure security policy for each wireless network:

a. [<u>specify the CA(s) from which the TSF will accept WLAN authentication server certificate(s)</u>]

b. ability to specify security type

c. ability to specify authentication protocol

d. specify the client credentials to be used for authentication

e. [***no additional functions***]

8. transition to the locked state

9. full wipe of protected data

10. configure application installation policy by

[<u>c. denying installation of applications</u>],

11. import keys/secrets into the secure key storage,

12. destroy imported keys/secrets and [<u>no other keys/secrets</u>] in the secure key storage,

13. import X.509v3 certificates into the Trust Anchor Database,

14. remove imported X.509v3 certificates and [<u>no other X.509v3 certificates</u>] in the Trust Anchor Database,

15. enroll the TOE in management

16. remove applications

17. update system software

18. install applications

[<u>20. enable/disable [**personal hotspot connections and tethered connections**]</u>

<u>22. enable data-at rest protection,</u>

<u>23. enable removable media's data-at-rest protection,</u>

<u>30. remove Enterprise applications,</u>

<u>42. [***enable/disable location services***]</u>]

6.3.4.3 FMT_SMF_EXT.1 Specification of Remediation Actions

FMT_SMF_EXT.1

The TSF shall offer [<u>full wipe of protected data, remove Enterprise applications, [***disable CC mode***]</u>] upon unenrollment and [<u>no other triggers</u>].

6.3.5 Class FPT: Protection of the TSF

6.3.5.1 FPT_AEX_EXT.1 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.

6.3.5.2 FPT_AEX_EXT.2 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.

6.3.5.3 FPT_AEX_EXT.3 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.

6.3.5.4 FPT_AEX_EXT.4 Domain Isolation

FPT_AEX_EXT.4.1

The TSF shall protect itself from modification by untrusted subjects.

FPT_AEX_EXT.4.2

The TSF shall enforce isolation of address space between applications.

6.3.5.5 FPT_KST_EXT.1 Key Storage

FPT_KST_EXT.1.1

The TSF shall not store any plaintext key material in readable non-volatile memory.

6.3.5.6 FPT_KST_EXT.2 No Key Transmission

FPT_KST_EXT.2.1

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

6.3.5.7 FPT_KST_EXT.3 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.

6.3.5.8 FPT_NOT_EXT.1 Event Notification

FPT_NOT_EXT.1.1

The TSF shall transition to non-operational mode and [no other actions] when the following types of failures occur:

- failures of the self tests
- TSF software integrity verification failures
- [no other failures].

6.3.5.9 FPT_STM.1 Reliable time stamps

FPT_STM.1.1

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

6.3.5.10 FPT_TST_EXT.1 TSF Cryptographic Functionality Testing

FPT_TST_EXT.1.1

The TSF shall run a suite of self-tests during initial start-up (on power on) to demonstrate the correct operation of all cryptographic functionality.

6.3.5.11 FPT_TST_EXT.2 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 [***no other executable code***], stored in mutable media prior to its execution through the use of [***a digital signature using a hardware-protected asymmetric key***].

6.3.5.12 FPT_TUD_EXT.1 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.

FPT_TUD_EXT.1.3

The TSF shall provide authorized users the ability to query the current version of installed mobile applications.

6.3.5.13 FPT_TUD_EXT.2 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.

FPT_TUD_EXT.2.2

The boot integrity [key, hash] shall only be updated by [verified software].

FPT_TUD_EXT.2.3

The digital signature verification key shall [match a hardware-protected public key].

FPT_TUD_EXT.2.4

The TSF shall verify mobile application software using a digital signature mechanism prior to installation.

6.3.6 Class FTA: TOE Access

6.3.6.1 FTA_SSL_EXT.1 TSF- and User-initiated locked state

FTA_SSL_EXT.1.1

The TSF shall transition to a locked state after a time interval of inactivity 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) [***no other actions***]

6.3.6.2 FTA_WSE_EXT.1 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.

6.3.7 Class FTP: Trusted Path/Channels

6.3.7.1 FTP_ITC_EXT.1 Trusted channel Communication

FTP_ITC_EXT.1.1

The TSF shall use 802.11-2012, 802.1X, and EAP-TLS and [IPsec] 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.

FTP_ITC_EXT.1.2

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

FTP_ITC_EXT.1.3

The TSF shall initiate communication via the trusted channel for connection to a wireless access point and [***no other communications***].

6.4 Statement of Security Functional Requirements Consistency

The Security Functional Requirements included in the ST represent all required SFRs specified in the PPs against which strict conformance is claimed and a subset of the optional SFRs. All hierarchical relationships, dependencies, and unfulfilled dependency rationales in the ST are considered to be identical to those that are defined in the claimed PP.

7 Security Assurance Requirements

This section identifies the Security Assurance Requirements (SARs) that are claimed for the TOE. The SARs which are claimed are consistent with the SARs that are defined in the claimed Protection Profile.

7.1 Class ADV: Development

7.1.1 Basic Functional Specification (ADV_FSP.1)

7.1.1.1 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.

7.1.1.2 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.

7.1.1.3 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.

7.2 Class AGD: Guidance Documentation

7.2.1 Operational User Guidance (AGD_OPE.1)

7.2.1.1 Developer action elements:

AGD_OPE.1.1D

The developer shall provide operational user guidance.

7.2.1.2 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.

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.

7.2.1.3 Evaluator action elements:

AGD_OPE.1.1E

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

7.2.2 Preparative Procedures (AGD_PRE.1)

7.2.2.1 Developer action elements:

AGD_PRE.1.1D

The developer shall provide the TOE including its preparative procedures.

7.2.2.2 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.

7.2.2.3 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.

7.3 Class ALC: Life Cycle Support

7.3.1 Labeling of the TOE (ALC_CMC.1)

7.3.1.1 Developer action elements:

ALC_CMC.1.1D

The developer shall provide the TOE and a reference for the TOE.

7.3.1.2 Content and presentation elements:

ALC_CMC.1.1C

The TOE shall be labeled with its unique reference.

7.3.1.3 Evaluator action elements:

ALC_CMC.1.1E

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

7.3.2 TOE CM Coverage (ALC_CMS.1)

7.3.2.1 Developer action elements:

ALC_CMS.1.1D

The developer shall provide a configuration list for the TOE.

7.3.2.2 Content and presentation elements:

ALC_CMS.1.1C

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

ALC_CMS.1.2C

The configuration list shall uniquely identify the configuration items.

7.3.2.3 Evaluator action elements:

ALC_CMS.1.1E

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

7.3.3 Timely Security Updates (ALC_TSU_EXT)

7.3.3.1 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.

7.3.3.2 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.

ALC_TSU_EXT.1.2C

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

ALC_TSU_EXT.1.3C

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

7.3.3.3 Evaluator action elements:

ALC_TSU_EXT.1.1E

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

7.4 Class ASE: Security Target Evaluation

7.4.1 Conformance Claims (ASE_CCL.1)

7.4.1.1 Developer action elements:

ASE_CCL.1.1D

The developer shall provide a conformance claim.

ASE_CCL.1.1D

The developer shall provide a conformance claim rationale.

7.4.1.2 Content and presentation elements:

ASE_CCL.1.1C

The conformance claim shall contain a CC conformance claim that identifies the version of the CC to which the ST and the TOE claim conformance.

ASE_CCL.1.2C

The CC conformance claim shall describe the conformance of the ST to CC Part 2 as either CC Part 2 conformant or CC Part 2 extended.

ASE_CCL.1.3C

The CC conformance claim shall describe the conformance of the ST to CC Part 3 as either CC Part 3 conformant or CC Part 3 extended.

ASE_CCL.1.4C

The CC conformance claim shall be consistent with the extended components definition.

ASE_CCL.1.5C

The conformance claim shall identify all PPs and security requirement packages to which the ST claims conformance.

ASE_CCL.1.6C

The conformance claim shall describe any conformance of the ST to a package as either package-conformant or package-augmented.

ASE_CCL.1.7C

The conformance claim rationale shall demonstrate that the TOE type is consistent with the TOE type in the PPs for which conformance is being claimed.

ASE_CCL.1.8C

The conformance claim rationale shall demonstrate that the statement of the security problem definition is consistent with the statement of the security problem definition in the PPs for which conformance is being claimed.

ASE_CCL.1.9C

The conformance claim rationale shall demonstrate that the statement of security objectives is consistent with the statement of security objectives in the PPs for which conformance is being claimed.

ASE_CCL.1.10C

The conformance claim rationale shall demonstrate that the statement of security requirements is consistent with the statement of security requirements in the PPs for which conformance is being claimed.

7.4.1.3 Evaluator action elements:

ASE_CCL.1.1E

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

7.4.2 Extended Components Definition (ASE_ECD.1)

7.4.2.1 Developer action elements:

ASE_ECD.1.1D

The developer shall provide a statement if security requirements.

ASE_ ECD.1.2D

The developer shall provide an extended components definition.

7.4.2.2 Content and presentation elements:

ASE_ ECD.1.1C

The statement of security requirements shall identify all extended security requirements.

ASE_ ECD.1.2C

The extended components definition shall define an extended component for each extended security requirement**.**

ASE_ ECD.1.3C

The extended components definition shall describe how each extended component is related to the existing CC components, families, and classes.

ASE_ ECD.1.4C

The extended components definition shall use the existing CC components, families, classes, and methodology as a model for presentation.

ASE_ ECD.1.5C

The extended components shall consist of measurable and objective elements such that conformance or nonconformance to these elements can be demonstrated.

7.4.2.3 Evaluator action elements:

ASE_ ECD.1.1E

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

ASE_ ECD.1.2E

The evaluator shall confirm that no extended component can be clearly expressed using existing components.

7.4.3 ST Introduction (ASE_INT.1)

7.4.3.1 Developer action elements:

ASE_INT.1.1D

The developer shall provide an ST introduction.

7.4.3.2 Content and presentation elements:

ASE_INT.1.1C

The ST introduction shall contain an ST reference, a TOE reference, a TOE overview and a TOE description.

ASE_INT.1.2C

The ST reference shall uniquely identify the ST.

ASE_INT.1.3C

The TOE reference shall identify the TOE.

ASE_INT.1.4C

The TOE overview shall summarize the usage and major security features of the TOE.

ASE_INT.1.5C

The TOE overview shall identify the TOE type.

ASE_INT.1.6C

The TOE overview shall identify any non-TOE hardware/software/firmware required by the TOE.

ASE_INT.1.7C

The TOE description shall describe the physical scope of the TOE.

ASE_INT.1.8C

The TOE description shall describe the logical scope of the TOE.

7.4.3.3 Evaluator action elements:

ASE_ INT.1.1E

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

ASE_ INT.1.2E

The evaluator shall confirm that the TOE reference, the TOE overview, and the TOE description are consistent with each other.

7.4.4 Security Objectives for the Operational Environment (ASE_OBJ.1)

7.4.4.1 Developer action elements:

ASE_ OBJ.1.1D

The developer shall provide a statement of security objectives.

7.4.4.2 Content and presentation elements:

ASE_ OBJ.1.1C

The statement of security objectives shall describe the security objectives for the operational environment.

7.4.4.3 Evaluator action elements:

ASE_ OBJ.1.1E

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

7.4.5 Stated Security Requirements (ASE_REQ.1)

7.4.5.1 Developer action elements:

ASE_ REQ.1.1D

The developer shall provide a statement of security requirements.

ASE_ REQ.1.2D

The developer shall provide a security requirements rational

7.4.5.2 Content and presentation elements:

ASE_ REQ.1.1C

The statement of security requirements shall describe the SFRs and the SARs.

ASE_ REQ.1.2C

All subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs shall be defined.

ASE_ REQ.1.3C

The statement of security requirements shall identify all operations on the security requirements.

ASE_ REQ.1.4C

All operations shall be performed correctly.

ASE_ REQ.1.5C

Each dependency of the security requirements shall either be satisfied, or the security requirements rationale shall justify the dependency not being satisfied.

ASE_ REQ.1.6C

The statement of security requirements shall be internally consistent.

7.4.5.3 Evaluator action elements:

ASE_ REQ.1.1E

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

7.4.6 TOE Summary Specification (ASE_TSS.1)

7.4.6.1 Developer action elements:

ASE_ TSS.1.1D

The developer shall provide a TOE summary specification.

7.4.6.2 Content and presentation elements:

ASE_ TSS.1.1C

The TOE summary specification shall describe how the TOE meets each SFR.

7.4.6.3 Evaluator action elements:

ASE_ TSS.1.1E

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

ASE_ TSS.1.2E

The evaluator shall confirm that the TOE summary specification is consistent with the TOE overview and the TOE description.

7.5 Class ATE: Tests

7.5.1 Independent Testing - Conformance (ATE_IND.1)

7.5.1.1 Developer action elements:

ATE_IND.1.1D

The developer shall provide the TOE for testing.

7.5.1.2 Content and presentation elements:

ATE_IND.1.1C

The TOE shall be suitable for testing.

7.5.1.3 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.

7.6 Class AVA: Vulnerability Assessment

7.6.1 Vulnerability Survey (AVA_VAN.1)

7.6.1.1 Developer action elements:

AVA_VAN.1.1D

The developer shall provide the TOE for testing.

7.6.1.2 Content and presentation elements:

AVA_VAN.1.1C

The TOE shall be suitable for testing.

7.6.1.3 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.

8 TOE Summary Specification

The following sections identify the security functions of the TOE and describe how the TSF meets each claimed SFR.

8.1 Cryptographic Support

8.1.1 FCS_CKM.1(1):

The TOE has CVL algorithm certificate #10 and ECDSA algorithm certificate #264 for Elliptic Curve key establishment and key generation respectively. The TOE has RSA algorithm certificate #960 for RSA key generation and for RSA key establishment, the TOE generally fulfills all of the NIST SP 800-56A and 800-56B requirements without extensions.

8.1.2 FCS_CKM.1(2):

The TOE supports asymmetric key generation of RSA and ECDSA (for curves P-256, P-384, and P-521) key pairs for signature generation and verification, which the TOE generates as per the ANSI X9.31 and FIPS 186-2, respectively. In particular, the TOE is compliant with section A.2.4 of ANSI X9.31-1998 and the key generation algorithm is implemented exactly as prescribed in this section. The TOE has RSA certificate #960 and ECDSA certificate # 264 for its RSA and ECDSA key generation implementations. While the TOE itself does not currently utilize its key generation implementation to generate any signature key pairs, the TOE provides key generation as a service to mobile applications executing on the TOE. The TOE allows applications to generate key pairs with any security strength, including those equal to or greater than 112-bits.

8.1.3 FCS_CKM.1(3):

The TOE adheres to 802.11-2012 with regards to 802.11i key generation. The TOE provides the PRF384 for WPA2, generates AES 128-bit keys using the OpenSSL module. The TOE has been designed for compliance, and the Devices have successfully completed certification (including WPA2 Enterprise) and received a WiFi CERTIFIED Interoperability Certificate from the WiFi Alliance (Certificate WFA54413). The WiFi Alliance maintains a website providing further information about the testing program: http://www.wi-fi.org/certification

8.1.4 FCS_CKM.2:

The TOE adheres to the RFC 3394, SP 800-38F, and 802.11-2012 standards and unwraps the GTK (sent wrapped using AES Key Wrap in an EAPOL-Key frame), thus ensuring that it does not expose the Group Temporal Key (GTK).

8.1.5 FCS_CKM_EXT.1:

The TOE supports a Root Encryption Key (REK) within the main (application) processor. Requests for encryption or decryption chaining to the REK are only accessible through the Trusted Execution Environment, or TEE (TrustZone). The REK lies in a series of 256-bit fuses, which the TOE programs during manufacturing. The TEE does not allow direct access to the REK but provides access to a derived HEK (Hardware Encryption Key) which is derived from the REK through a KDF function for encryption and decryption.

The REK value is generated during manufacturing by the TOE (if it detects that the REK fuses have not been set) using its hardware DRBG.

8.1.6 FCS_CKM_EXT.2:

The TOE supports Data Encryption Key (DEK) generation using its approved RBGs. The TOE generates AES 256-bit DEKs using its OpenSSL CTR_DRBG in response to an application through either the Tizen Web API or through the native C API. This is used, for example, by the ODE (Onboard Device Encryption, which refers to the TOE's encryption of the user data partition) system using the native C API as well as by the media encryption system application to generate a master key for protection of File Encryption Keys (FEKs) for individual files.

The REK value is generated during manufacturing by the TOE (if it detects that the REK fuses have not been set) using its hardware DRBG.

8.1.7 FCS_CKM_EXT.3:

The TOE generates KEKs (which are always AES 256-bit keys generated by the CTR_DRBG) through a combination of methods. First, the module generates a KEK (denoted as a Domain KEK, or DKEK) for each user of the phone, for each Domain (where a Domain provides separation of application data, e.g., "Corporate" or "Personal"). Note that the current TOE only supports a single Domain for each user. However, future versions of the TOE may support multiple Domains.

8.1.8 FCS_CKM_EXT.4:

The TOE destroys cryptographic keys when they are no longer in use by the system. The exceptions to this are public keys (that protect the boot chain and software updates) and the REK which, is never cleared. Keys that are stored in RAM for immediate use are destroyed by a zero overwrite. Keys which are stored in Flash (i.e. eMMC) are destroyed by cryptographic erasure through a Secure Trim command (as defined in the JEDEC eMMC 5.0 specification, JESD84-B50) to the flash controller for the location where the ODE and removable media keys are stored. Once these are erased, the keys that are stored within the encrypted data partition of the TOE are considered cryptographically erased.

8.1.9 FCS_CKM_EXT.5:

The TOE provides a TOE Wipe function that first erases the ODE DEK used to encrypt the entire data partition using a Secure Trim command to ensure that TOE writes zeros to the eMMC blocks containing the ODE DEK (the data partition footer contains the ODE DKEK and ODE DEK as well as the SDCard DKEK and SDCard MK). After performing this key erasing, the TOE will also format the partition by resetting the partition table and disk references are reset (note due to the erasing of the keys these references cannot be used to retrieve any data). When the reformatting of the partition is complete, the TOE will perform a power-cycle.

8.1.10 FCS_CKM_EXT.6:

The TOE creates salts and nonces (salt values used in WPA2) using its AES-256 CTR_DRBG.

8.1.11 FCS_COP.1(1):

The TOE performs cryptographic algorithms in accordance with the following NIST standards and has received the following CAVP algorithm certificates.

The OpenSSL FIPS Object Module provides the following algorithms:

Algorithm	NIST Standard	SFR Reference	Cert#
AES 128/256 CBC, CCM, GCM, KW	FIPS 197, SP 800-38A/C/D/F	FCS_COP.1(1)	1884
CVL ECC CDH P-224/256/384/521	SP 800-56A	FCS_CKM.1(1)	10
DRBG Hash/HMAC/CTR/Dual_EC	SP 800-90A	FCS_RBG_EXT.1(1)	157
ECDSA PKG/PKV/SigGen/SigVer	FIPS 186-4	FCS_CKM.1(1) FCS_CKM.1(2) FCS_COP.1(3)	264
HMAC SHA-1/256/384/512	FIPS 198-1 & 180-4	FCS_COP.1(4)	1126
RSA SIG(gen)/SIG(ver)/Key(gen)	FIPS 186-2	FCS_CKM.1(1) FCS_CKM.1(2) FCS_COP.1(3)	960
SHS SHA-1/256/384/512	FIPS 180-4	FCS_COP.1(2)	1655

Table 8-1: OpenSSL Cryptographic Algorithms

The TOE performs cryptographic algorithms in accordance with the following NIST standards and has received the following CAVP algorithm certificates.

Algorithm	NIST Standard	SFR Reference	Cert#
AES 128/256 CBC	FIPS 197, SP 800-38A	FCS_COP.1(1)	2810 & 2809
HMAC SHA-1/256/384/512	FIPS 198-1 & 180-4	FCS_COP.1(4)	1761 & 1760
RNG ANSI X9.31 AES-128	ANSI X9.31 & 931rngext.pdf	FCS_RBG_EXT.1(2)	1276 & 1275
SHS SHA-1/256/384/512	FIPS 180-4	FCS_COP.1(2)	2358 & 2357

Table 8-2: Security Kernel Cryptographic Algorithms

The TOE's application processors include hardware entropy implementations that supply random data within the TEE and to the Linux kernel RNG (/dev/random).

Note that kernel-space system applications utilize the cryptographic algorithm implementations in the Samsung Kernel Cryptographic Module (Kernel Crypto), while user-space system applications and mobile applications utilize the OpenSSL module (either through a Web API or through the native C API). In the case of each cryptographic module, the module itself includes any algorithms required (for example, OpenSSL provides hash functions for use by HMAC and digital signature algorithms).

8.1.12 FCS_COP.1(2):

See FCS_COP.1(1), "Table 8-1 OpenSSL Cryptographic Algorithms" and Table 8-2 Samsung Kernel Cryptographic Algorithms. The TOE supports all SHA sizes save 224 (e.g., SHA-1, 256, 384, & 512).

8.1.13 FCS_COP.1(3):

See FCS_COP.1(1), "Table 8-1 OpenSSL Cryptographic Algorithms".

8.1.14 FCS_COP.1(4):

For its HMAC implementations, the TOE accepts all key sizes of 160, 256, 384, & 512; supports all SHA sizes save 224 (e.g., SHA-1, 256, 384, & 512), utilizes the specified block size (512 for SHA-1 and 256, and 1024 for SHA-384 & 512), and output MAC lengths of 160, 256, 384, and 512.

8.1.15 FCS_COP.1(5):

The TOE conditions the user's password exactly as per SP 800-132 (and thus as per SP 800-197-1) with no deviations by using PBKDF2 with 16,384 HMAC-SHA-512 iterations to combine a 128-bit salt with the user’s password.

8.1.16 FCS_HTTPS_EXT.1:

The TOE includes the ability to support the HTTPS protocol (compliant with RFC 2818) so that (mobile and system client) applications executing on the TOE can securely connect to external servers using HTTPS. Administrators have no credentials and cannot use HTTPS or TLS to establish administrative sessions with the TOE as the TOE does not provide any such capabilities.

8.1.17 FCS_IV_EXT.1:

The TOE generates IVs for data storage encryption and for key storage encryption. The TOE uses AES-CBC mode for data encryption and AES-GCM for key storage.

8.1.18 FCS_RBG_EXT.1:

The TOE provides a number of different RBGs including

1. A SHA-256 Hash_DRBG provided in the hardware of the Qualcomm Application Processor.
2. RBGs provided by OpenSSL: Hash_DRBG (using any size SHA), HMAC_DRBG (again using size SHA), and CTR_DRBG (using AES). Note that while the TOE includes implementations of three DRBG variants (and supports all options within each variant), the TOE (and its current system level applications) make use of only an AES-256 CTR_DRBG. Furthermore The TOE provides mobile applications access (through an Tizen Web API) to random data drawn from its AES-256 CTR_DRBG. However, future (system-level) applications could utilize other DRBG variants supported by the TOE’s OpenSSL module.
3. And an ANSI X9.31 AES-128 RBG provided by Kernel Crypto

The TOE ensures that it initializes each RBG with sufficient entropy which is accumulated from a TOE-hardware-based noise source (please see the Entropy Assessment Report for more details). The TOE uses its hardware-based noise source to continuously fill /dev/random with random data with full entropy, and in turn, the TOE draws from /dev/random to seed both its AES-256 CTR_DRBG and ANSI X9.31 AES-128 RBG. The TOE seeds its AES-256 CTR_DRBG using 384-bits of data from /dev/random, thus ensuring at least 256-bits of entropy. Finally the TOE seeds its X9.31 AES-128 with 256-bits of data from /dev/random, also ensuring that it contains at least 128-bits of entropy.

8.1.19 FCS_SRV_EXT.1:

The TOE provides applications access to the cryptographic operations including encryption (AES), hashing (SHA), signing and verification (RSA & ECDSA), key hashing (HMAC), password-based key-derivation functions (PKBDFv2 HMAC-SHA-512), generate asymmetric keys for key establishment

(RSA, DH, and ECDH), and generate asymmetric keys for signature generation and verification (RSA, ECDSA). The TOE provides access through the Tizen operating system's Web API, through the native OpenSSL API, and through the kernel. The vendor also developed testing applications to enable execution of the NIST algorithm testing suite in order to verify the correctness of the algorithm implementations.

8.1.20 FCS_STG_EXT.1:

The TOE provides for secure key storage for asymmetric keys. For these asymmetric keys, the TOE secures a series of public keys used for Secure Boot through a chain of trust that operates as follows. The Application Processor (AP) contains the SHA-256 hash of the Secure Boot Public Key (an RSA 2048-bit key embedded in end of the signed bootloader image), and upon verifying the SBPK attached to the bootloader produces the expected hash, the AP uses this public key to verify the PKCS 1.5 RSA 2048 w/ SHA-256 signature of the bootloader image, to ensure its integrity and authenticity before transitioning execution to the bootloader. The bootloader, in turn, contains the Image Signing Public Key (ISPK), which the bootloader will use to verify the PKCS 1.5 RSA 2048 w/ SHA-1 signature on either kernel image (primary kernel image or recovery kernel image). Note that when configured for Common Criteria mode, the TOE only accepts updates to the TOE firmware Over The Air; however, when not configured for CC mode, the TOE allows updates through the bootloader's ODIN mode. The primary kernel includes an embedded FOTA Public Key, which the TOE uses to verify the authenticity and integrity of Firmware Over-The-Air update signatures (which contain a PKCS 2.1 PSS RSA 2048 w/ SHA-512 signature).

In addition to the internal public keys protected by the chain-of-trust with a hardware root/anchor, the TOE also provides users and applications running on the TOE the ability to import keys. The TOE allows a user to import a certificate (in PKCS#12 [PFX] format) and provides applications running on the TOE an API to import a certificate. In either case, the TOE will place the certificate into the user's key store, where the TOE will remove the PKCS#12 password-based protection from the certificate and then encrypt the certificate with a DEK, which in turn is encrypted with the a KEK derived from the user's DKEK. All user and application items placed into the user's keystore are secured in this fashion. Note that while operating in CC mode, the TOE also encrypts (using ODE) the Flash filesystem in which keystore items reside, providing a second layer of encryption protection to keystore objects.

The user of the TOE can elect to remove values from the keystore, as well as to securely wipe the entire device.

The TOE provides applications control (control over use and destruction) of keys that they create or import, and only the common application developer can explicitly authorize access, use, or destruction of one application's key by any other application.

8.1.21 FCS_STG_EXT.2:

The TOE provides protection for all stored keys (i.e. those written to storage media for persistent storage) chained to both the user's password and the REK. FEKs (which are DEKs for individual encrypted files on removable media), which are 128-bit, and stored as part of the metadata for each encrypted file, are encrypted with AES-CBC. The FEK is protected by the SDCard MK, a 256-bit key. All other keys (including the SDCard MK) are encrypted with AES-GCM. All KEKs are 256-bit, ensuring that that the TOE encrypts every key with another key of equal or greater strength/size.

In the case of WiFi, the TOE utilizes the 802.11-2012 KCK and KEK keys to unwrap (decrypt) the WPA2 Group Temporal Key received from the Access Point. Additionally, the TOE protects persistent Wifi keys (user certificates and Pre-Shared Key values) in the same way as other DEKs, namely by encrypting them with a KEK chained to both the user's password and the REK.

8.1.22 FCS_STG_EXT.3:

The key hierarchy shows AES-256 GCM is used to encrypt all KEKs while the GCM encryption mode itself ensures integrity as authenticated decryption operations fail if the encrypted KEK becomes corrupted.

8.1.23 FCS_TLS_EXT.1:

The TOE supports EAP-TLS using TLS version 1.0 and supports the following ciphersuites:

TLS_RSA_WITH_AES_128_CBC_SHA

TLS_RSA_WITH_AES_256_CBC_SHA

TLS_DHE_RSA_WITH_AES_128_CBC_SHA

TLS_DHE_RSA_WITH_AES_256_CBC_SHA

The user or administrator does not need to configure (nor is it possible to configure) the TOE in any special way to ensure that the TOE exclusively uses (when configured for CC mode) the above set of TLS ciphersuites as part of EAP-TLS.

When configuring the TOE to utilize EAP-TLS as part of a WPA2 protected WiFi-network, the user must explicitly select the CA certificate to which the server's certificate must chain. Moreover, the user (or administrator, using an MDM Server), must load such a CA certificate as the TOE does not come with any built-in CA certificates suitable for use with 802.1X.

8.1.24 FCS_TLS_EXT.2:

The TOE provides mobile applications API service for TLS versions 1.0, 1.1, and 1.2 including support for following twelve ciphersuites.

- TLS_RSA_WITH_AES_128_CBC_SHA
- 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

If an application detects that the distinguished name (DN) contained within the peer certificate does not match the expected DN for the peer when using the provided APIs in an attempt to establish a trusted channel, then the application should not establish the connection.

8.2 User Data Protection

8.2.1 FDP_ACF_EXT.1:

Every application is signed with a signature that defines its level of access, which are either: Public, Platform or Partner.

Public - Applications are allowed access to system services based on user authorization. Prior to downloading the application the user must accept the system services requested by the application for its use. The signature's validity is checked but it can be signed by any third-party (i.e. the developer).

Partner - Applications are allowed access to system services based on user authorization. Prior to downloading the application the user must accept the system services requested by the application for its use. The signature must be Platform or Partner level.

Platform - Applications are allowed access to system services based on user authorization when downloaded. Prior to downloading the application the user must accept the system services requested by the application for its use. Built-in applications do not require user authorization to system services. The signature must be Platform level.

The Partner level signature is provided by the Tizen Store and the Platform level signature is provided by Samsung. For the Public level, the developer's or the App Store's signature is provided.

A list of privileged system services can be found here:

https://wiki.tizen.org/wiki/Security/Tizen_2.X_Privileges

8.2.2 FDP_DAR_EXT.1:

The TOE provides AES-256 CBC encryption of all data (which includes both user data and TSF data) stored on the TOE through On Device Encryption (ODE) of the data partition. The TOE also has TSF data for its ODE keystore, which the TOE stores outside the ODE encrypted data partition, and the TOE also includes a read-only filesystem in which the TOE's system executables, libraries, and their configuration data reside. For its ODE encryption of the data partition (where the TOE stores all user data and all application data), the TOE uses an AES-256 bit DEK with CBC feedback mode to encrypt the entire partition. The TOE also provides AES-128 CBC encryption of protected data stored on the external SDCard using FEKs. The TOE encrypts each individual file stored on the SDCard, generating a unique FEK for each file.

8.2.3 FDP_STG_EXT.1(1):

The TOE's Trust Anchor Database consists of the built-in certs (individually stored in /system/etc/security/ and which can be disabled through the Tizen UI)) and additional user loaded certs. The built-in certs are protected since they are read only, and part of the lower-level system, but the user-loaded certs are protected by the keystore.

8.3 Identification and Authentication

8.3.1 FIA_AFL_EXT.1:

The TOE maintains, for each user, the number of failed logins since the last successful login, and upon reaching the maximum number of incorrect logins, the TOE performs a full wipe of all protected data (and in fact, wipes all users' data). An administrator can adjust the failed login threshold from the default of ten failed logins to a value defined through an MDM Server between one and ninety-nine.

8.3.2 FIA_PAE_EXT.1:

The TOE can join WPA2-802.1X (802.11i) wireless networks requiring EAP-TLS authentication, acting as a client/supplicant (and in that role connect to the 802.11 access point and communicate with the 802.1X authentication server).

8.3.3 FIA_PMG_EXT.1:

The TOE authenticates user through a password consisting of upper and lower case letters, numbers, and the following special characters:

!	@	#	$	%	^	&	*	(	)

By default the TOE requires passwords to have a minimum of six characters but no more than sixteen characters, contain at least one letter and contain at least one number. However, an MDM Server can change these defaults on the TOE.

8.3.4 FIA_TRT_EXT.1:

The TOE does not provide any method for authentication through an external port, meaning a user must authenticate through the standard User Interface. Additionally, the TOE limits the number of authentication attempts through the UI to no more than five attempts within 30 seconds (thus if the current [the nth] and prior four authentication attempts have failed, and the n-4th attempt was less than 30 second ago, the TOE will prevent any further authentication attempts until thirty seconds have elapsed since that n-4th attempt).

8.3.5 FIA_UAU.7:

The TOE allows the user to enter the user's password from the lock screen. The TOE will, by default, display the most recently entered character of the password briefly or until the user enters the next character in the password, at which point the TOE obscures the character by replacing the character with a dot symbol.

8.3.6 FIA_UAU_EXT.1:

As detailed previously, the TOE's Key Hierarchy requires the user's password in order to derive the PKEK1 & PKEK2 keys in order to decrypt other KEKs and DEKs. Thus, until it has the user's password, the TOE cannot decrypt the DEK utilized by ODE to decrypt protected data.

8.3.7 FIA_UAU_EXT.2:

The TOE, when configured to require a user password (as is the case in CC mode), allows a user to make an emergency call, take screen shots (stored internally), turn the TOE off, enable or disable airplane mode, enable Bluetooth or place an outgoing phone call to the last incoming call immediately after the last incoming call is disconnected. Other than those actions, a user cannot perform any other actions other than observing notifications displayed on the lock screen until after successfully authenticating.

8.3.8 FTA_SSL_EXT.1 FIA_UAU_EXT.3:

The TOE requires the user to enter their password in order to unlock the TOE. Additionally the TOE requires the user to confirm their current password when accessing the "Settings->Screen Security" menu in the TOE's user interface. Only after entering their current user password can the user then elect to change their password.

8.3.9 FIA_X509_EXT.1:

The TOE checks the validity of all imported CA certificates by checking for the presence of the basicConstraints extension and that the CA flag is set to TRUE as the TOE imports the certificate. Additionally the TOE verifies the extendedKeyUsage Server Authentication purpose during WPA2/EAP-TLS negotiation**.** Certificates used for trusted updates and 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. The TOE enforces RFC 5280 certificate validation and certificate path validation with the certificate path terminating with a certificate in the Trust Anchor Database. The TOE also validates the revocation status of certificates using the Online Certificate Status Protocol (OCSP) as specified in RFC 2560.

8.3.10 FIA_X509_EXT.2:

The TOE uses X.509v3 certificates as part of EAP-TLS and IPsec authentication. The TOE comes with a built-in set of default of Trusted Credentials (Tizen's set of trusted CA certificates). While the user cannot remove any of the built-in default CA certificates, the user can disable any of those certificates through the user interface so that certificates issued by disabled CA's cannot validate successfully. In addition, a user can import a new trusted CA certificate into the Key Store or an administrator can install a new certificate through an MDM Server.

The TOE does not establish TLS connections itself (beyond EAP-TLS used for WPA2 Wifi connections), but provides a series of APIs that mobile applications can use to check the validity of a peer certificate. After correctly using the specified APIs, the mobile device can be assured of the validity of the peer certificate and will not establish the trusted connection if the peer certificate cannot be verified (including validity, certification path, and revocation through OCSP). The exception to this is in the use of EAP-TLS for wireless communications, which will not be able to verify a revoked certificate because the TOE is not yet connected to the wireless network at the time the certificate checking occurs.

8.3.11 FIA_X509_EXT.3:

The TOE's Tizen operating system provides applications the certsvc_certificate_verify_with_caflag and certsvc_ocsp_check APIs for validating certification paths (certificate chains) establishing a trust chain from a certificate to a trust anchor.

8.4 Security Management

8.4.1 FMT_MOF.1(1):

The TOE provides the following management functions which may only be performed by the user:

1. enroll the TOE in management
2. enable/disable the VPN protection

5. enable/disable personal Hotspot connections and tethered connections

8.4.2 FMT_MOF.1(2):

The TOE restricts the ability to perform the following 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 camera and microphone

4. configure application installation policy by

    c. denying installation of applications

6. enable/disable Bluetooth and WiFi

31. enable/disable location services

8.4.3 FMT_SMF.1:

The TOE provides 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 Bluetooth and WiFi

5. enable/disable camera and microphone

6. specify wireless networks (SSIDs) to which the TSF may connect

7. configure security policy for each wireless network:

a. specify the CA(s) from which the TSF will accept 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

8. transition to the locked state

9. full wipe of protected data

10. configure application installation policy by

c. denying installation of applications,

11. import keys/secrets into the secure key storage

12. destroy imported keys/secrets and no other keys/secrets in the secure key storage

13. import X.509v3 certificates into the Trust Anchor Database

14. remove imported X.509v3 certificates and all other X.509v3 certificates in the Trust Anchor Database,

15. enroll the TSF in management

16. remove applications

17. update system software

18. install applications

20. enable/disable personal hotspot connections and tethered connections

22. enable data-at rest protection

23. enable removable media’s data-at-rest protection

30. remove Enterprise applications

42. enable/disable location services

8.4.4 FMT_SMF_EXT.1:

When unenrolled from an MDM Server in the evaluated configuration, the TOE will enforce the MDM Server policies for unenrollment which includes performing a full wipe of protected data, removing Enterprise applications and the disabling CC mode of operation.

8.5 Protection of the TSF

8.5.1 FPT_AEX_EXT.1:

The Linux kernel of the TOE’s Tizen operating system provides address space layout randomization utilizing a non-cryptographic kernel random function to provide 8 unpredictable bits to the base address of any user-space memory mapping. The random function, though not cryptographic, ensures that one cannot predict the value of the bits.

8.5.2 FPT_AEX_EXT.2:

The TOE's Tizen 2.3 operating system utilizes a 3.4 Linux kernel, whose memory management unit (MMU) enforces read, write, and execute permissions on all pages of virtual memory. The Tizen operating system (as of Tizen 2.3) sets the ARM No eXecute (XN) bit on memory pages and the TOE’s ARMv7 Application Processor’s MMU circuitry enforces the XN bits. From Tizen’s documentation (https://source.Tizen.com/devices/tech/security/index.html), Tizen 2.3 forward supports “Hardware-based No eXecute (NX) to prevent code execution on the stack and heap”.

8.5.3 FPT_AEX_EXT.3:

The TOE's Tizen operating system provides explicit mechanisms to prevent stack buffer overruns (enabling -fstack-protector) in addition to taking advantage of hardware-based No eXecute to prevent code execution on the stack and heap. Samsung requires these protections and applies them to all TSF executable binaries and libraries.

8.5.4 FPT_AEX_EXT.4:

The TOE protects itself from modification by untrusted subjects using a variety of methods. The first protection employed by the TOE is a Secure Boot process that uses cryptographic signatures to ensure the authenticity and integrity of the bootloader and kernels using data fused into the device processor.

The TOE protects access to the REK and derived HEK to only trusted applications within the TEE (TrustZone). The TOE key manager includes a TEE module which utilizes the HEK to protect all other keys in the key hierarchy. All TEE applications are cryptographically signed, and when invoked at runtime (at the behest of an untrusted application), the TEE will only load the trusted application after successfully verifying its cryptographic signature. Furthermore, SKMM_TA checks the integrity of the system by checking the result from both Secure Boot/Security Manager and from the Integrity Check Daemon before servicing any requests. Without the TEE application, no keys within the TOE (including keys for ScreenLock, the key store, and user data) can be successfully decrypted, and thus are useless.

The third protection is the TOE Security Manager watchdog service. The Security Manager manages the CC mode of the TOE by looking for unsigned kernels or failures from other, non-cryptographic checks on system integrity, and upon detecting of a failure in either, disables the CC mode and notifies the TEE application. The TEE application then locks itself, again rendering all TOE keys useless.

Finally, the TOE's Tizen OS provides "sandboxing" that ensures that each non-system mobile application executes with the file permissions of a unique Linux user ID, in a different virtual memory space. This ensures that applications cannot access each other's memory space (it is possible for two processes to utilize shared memory, but not directly access the memory of another application) or files and cannot access the memory space or files of system-level applications.

8.5.5 FPT_KST_EXT.1:

The TOE does not store any plaintext key material in its internal Flash or on external SDCards. It instead encrypts all keys before storing them. This ensures that irrespective of how the TOE powers down (e.g., a user commands the TOE to power down, the TOE reboots itself, or battery is removed), all keys in internal Flash or on an external SDCard are wrapped with a KEK. Please refer to section 8.1 of the TSS for further information (including the KEK used) regarding the encryption of keys stored in the internal Flash and on external SDCards. Since the TOE encrypts all keys stored in Flash, the TOE must first decrypt and utilize keys upon boot-up.

8.5.6 FPT_KST_EXT.2:

The TOE utilizes a cryptographic module consisting of an implementation of OpenSSL, the kernel crypto module, the key management module, and the following system-level executables that utilize KEKs: dm-crypt, eCryptfs, wpa_supplicant, and the keystore.

The TOE ensures that plaintext key material does not leave this cryptographic module. The functions described by FPT_KST_EXT.2, FCS_STG_EXT.1, FCS_STG_EXT.2, and FCS_STG_EXT.3 provide additional information on secure key storage in order to show how secure key storage is achieved.

8.5.7 FPT_KST_EXT.3:

The TOE does not provide any way to export plaintext DEKs or KEKs (including all keys stored in the keystore) as the TOE chains all KEKs to the HEK/REK.

8.5.8 FPT_NOT_EXT.1:

When the TOE encounters a self-test failure or when the TOE software integrity verification fails, the TOE transitions to a non-operational mode. The user may attempt to power-cycle the TOE to see if the failure condition persists, and if it does persist, the user may attempt to boot to the recovery mode/kernel to wipe data and perform a factory reset in order to recover the device.

8.5.9 FPT_STM.1:

The TOE requires time for the Package Manager, FOTA image verifier, wpa_supplicant, and key store applications. TOE uses the Cellular Carrier time as a trusted source; however, the user can also manually set the time through the TOE's user interface.

8.5.10 FPT_TST_EXT.1:

The TOE performs known answer power on self-tests (POST) on its cryptographic algorithms to ensure that they are functioning correctly. The kernel itself performs known answer tests on its cryptographic algorithms to ensure they are working correctly and the Security Manager service invokes self-tests of OpenSSL at start-up to ensure that those cryptographic algorithms are working correctly. In the event that a self-test fails, the TOE will attempt a reboot in an effort to clear the error.

Algorithm	Implemented In	Description
AES encryption/decryption	OpenSSL	Comparison of known answer to calculated valued
ECDH key agreement	OpenSSL	Comparison of known answer to calculated valued
DRBG random bit generation	OpenSSL	Comparison of known answer to calculated valued
ECDSA sign/verify	OpenSSL	Comparison of known answer to calculated valued
HMAC-SHA	OpenSSL	Comparison of known answer to calculated valued
RSA sign/verify	OpenSSL	Comparison of known answer to calculated valued
SHA hashing	OpenSSL	Comparison of known answer to calculated valued
AES encryption/decryption	Kernel Crypto	Comparison of known answer to calculated valued
HMAC-SHA	Kernel Crypto	Comparison of known answer to calculated valued
SHRNG random bit generation	Kernel Crypto	Comparison of known answer to calculated valued
SHA hashing	Kernel Crypto	Comparison of known answer to calculated valued

Table 8-3: Power-up cryptographic Algorithm Known Answer Tests

8.5.11 FPT_TST_EXT.2:

The TOE ensures a secure boot process in which the TOE verifies the digital signature of the bootloader software for the Application Processor (using a public key whose hash resides in the processor's internal fuses) before transferring control. The bootloader, in turn, verifies the signature of the Linux kernel (either the primary or the recovery kernel) it loads.

8.5.12 FPT_TUD_EXT.1:

The TOE's user interface provides a method to query the current version of the TOE software/firmware (Tizen version, baseband version, kernel version, and build number) and hardware (model and version). Additionally, the TOE provides users the ability to review the currently installed apps (including 3rd party "built-in" applications) and their version.

8.5.13 FPT_TUD_EXT.2:

When in CC mode, the TOE verifies all updates to the TOE software using a public key (FOTA public key) chaining ultimately to the Secure Boot Public Key (SBPK), a hardware protected key whose SHA-256 hash resides inside the application processor (note that when not in CC mode, the TOE allows updates to the TOE software through ODIN mode of the bootloader). After verifying an update's FOTA signature, the TOE will then install those updates to the TOE. The application processing verifies the bootloader's authenticity and integrity (thus tying the bootloader and subsequent stages to a hardware root of trust: the SHA-256 hash of the SBPK, which cannot be reprogrammed after the "write-enable" fuse has been blown).

The Tizen OS requires that all applications bear a valid signature before the TOE will install the application. When updating an application, the signature that is provided with the update must match the signature on the installed application or the TOE will not install the update.

8.6 Timely Security Updates

8.6.1 ALC_TSU_EXT.1:

Samsung utilizes industry best practices to ensure their devices are patched to mitigate security flaws. Samsung provides customers with a developer web portal (http://developer.samsung.com/notice/How-to-Use-the-Forum) in which one can ask questions as well as inform Samsung of potential issues with their software (as Samsung moderators monitor the forums), and as an Tizen OEM, also works with the Tizen Development Group on reported Tizen issues to ensure customer devices are secure.

Samsung will create updates and patches to resolve reported issues as quickly as possible, at which point the update is provided to the wireless carriers. The delivery time for resolving an issue depends on the severity, and can be as rapid as a few days before the carrier handoff for high priority cases. The wireless carriers perform additional tests to ensure the updates will not adversely impact their networks and then plan device rollouts once that testing is complete. Carrier updates usually take at least two weeks to as much as two months (depending on the type and severity of the update) to be rolled out to customers.

Samsung communicates with the reporting party to inform them of the status of the reported issue. Further information about updates is handled through the carrier release notes.

8.7 TOE Access

8.7.1 FTA_SSL_EXT.1:

The TOE transitions to its locked state either immediately after a User initiates a lock by pressing the power button or after a configurable period of inactivity (default of 1 minute), and as part of that transition, the TOE will display a lock screen to obscure the previous contents. Prior to authenticating to the TOE, a user cannot perform any actions other than make emergency call, take screenshots, receive calls, enable airplane mode, turn off the TOE, restart the TOE, enable Bluetooth or place an outgoing phone call to the last incoming call immediately after the last incoming call is disconnected.

Note that during power up, the TOE presents the user with an initial power-up ODE screen where the user can only make an emergency call or enter the ODE password in order to allow the TOE to decrypt the

ODE key so as to be able to access the data partition. After successfully authenticating at the power-up login screen, the TOE presents the user with the lock screen.

8.7.2 FTA_WSE_EXT.1:

The TOE allows an administrator to specify (through the use of an MDM Server) a list of wireless networks (SSIDs) to which the user may direct the TOE to connect to. When not enrolled with an MDM Server, the TOE allows the user to control to which wireless networks the TOE should connect, but does not provide an explicit list of such networks, rather the use may scan for available wireless network (or directly enter a specific wireless network), and then connect. Once a user has connected to a wireless network, the TOE will automatically reconnect to that network when in range and the user has enabled the TOE's WiFi radio.

8.8 Trusted Path/Channels

8.8.1 FTP_ITC.1:

The TOE provides secured (encrypted and mutually authenticated) communication channels between itself and other trusted IT products through the use of 802.11-2012, 802.1X, and EAP-TLS and IPsec. The TOE permits itself and applications to initiate communicate via the trusted channel, and the TOE initiates communicate via the trusted channel for connection to a wireless access point. The TOE has also been validated against the Protection Profile for IPsec Virtual Private Network (VPN) Clients.