Apple Inc.

Apple iPad and iPhone Mobile Devices with iOS 11.2

PP_MD_V3.1, EP_MDM_AGENT_V3.0, & PP_WLAN_CLI_EP_V1.0

Common Criteria Guide

Version 1.01
2018-03-30
VID: 10851

Prepared for:
Apple Inc.
One Apple Park Way
MS 927-1CPS
Cupertino, CA 95014
www.apple.com

Prepared by:
atsec information security Corp.
9130 Jollyville Road, Suite 260
Austin, TX 78759
www.atsec.com

Table of Contents

Table of Figures

Table of Tables

Revision History

Version	Date	Change
1.0	2018-02-26	Initial Version
1.01	2018-03-30	Changes in response to validators comments

1 Introduction

According to the “Apple iPad and iPhone Mobile Devices with iOS 11.2 PP_MD_V3.1, EP_MDM_AGENT_V3.0, & PP_WLAN_CLI_EP_V1.0 Security Target” ([ST]), the Target of Evaluation (TOE) is a series of Apple iPad and iPhone mobile devices running the iOS 11.2 operating system. The operating system manages the device hardware, provides mobile device agent functionality, and provides the technologies required to implement native applications (apps). iOS 11.2 provides a built-in mobile device management (MDM) application programming interface (API), giving management features that may be utilized by external MDM solutions and allowing enterprises to use profiles to control some of the device settings. The TOE provides a consistent set of capabilities allowing the supervision of enrolled devices. These capabilities include the preparation of devices for deployment, the subsequent management of the devices, and the termination of management.

The TOE does not include the user apps that run on top of the operating system but does include controls that limit application behavior. The TOE is expected but not required to be part of an MDM solution that enables the enterprise to control and administer all TOE instances that are part of the enterprise MDM solution.

1.1 Purpose

This document provides guidance on the secure installation and secure use of the TOE for the:

- [PP_MD_V3.1] U.S. Government Approved Protection Profile - Protection Profile for Mobile Device Fundamentals, Version 3.1 (https://www.niap-ccevs.org/Profile/Info.cfm?id=417);
- [EP_MDM_AGENT_V3.0] U.S. Government Approved Protection Profile - Extended Package for Mobile Device Management Agents Version 3.0 (https://www.niap-ccevs.org/Profile/Info.cfm?id=403); and
- [PP_WLAN_CLI_EP_V1.0] Extended Package for WLAN Client Version 1.0 (https://www.niap-ccevs.org/Profile/Info.cfm?id=386)

in the evaluated configuration according to the Apple iPad and iPhone Mobile Devices with iOS 11.2 PP_MD_V3.1, EP_MDM_AGENT_V3.0, & PP_WLAN_CLI_EP_V1.0 Security Target.

This document provides clarifications and changes to the Apple documentation and shall be used as the guiding document for the configuration and administration of the TOE in the Common Criteria (CC) evaluated configuration. The official Apple documentation should be referred to and followed only as directed within this guiding document. Table *1*: TOE Guidance Documents, lists the guidance documents relevant to the configuration and operation of the TOE.

Document Name	Location
User Guidance	
[iPhone_UG] iPhone User Guide for iOS 11 (2017)	https://help.apple.com/iphone/11/
[iPad_UG] iPad User Guide for iOS 11 (2017)	https://help.apple.com/ipad/11/
Administrator Guidance	

Document Name	Location
[CC_GUIDE] Apple iPad and iPhone Mobile Devices with iOS 11.2 PP_MD_V3.1, EP_MDM_AGENT_V3.0, & PP_WLAN_CLI_EP_V1.0 Common Criteria Guide	(This document.) https://www.niap-ccevs.org/st/st_vid10851-agd.pdf
Supporting Documents	
[iOSDeployRef] iOS Deployment Reference (V3.9)	https://itunes.apple.com/us/book/ios-deployment-reference/id917468024?mt=11
[OTA_Guide] Over-The-Air Profile Delivery and Configuration Guide (Updated 2018-01-24)	https://developer.apple.com/library/content/documentation/NetworkingInternet/Conceptual/iPhoneOTAConfiguration/Introduction/Introduction.html
[IOS_CFG] Configuration Profile Reference (Updated 2018-01-24)	https://developer.apple.com/enterprise/ConfigurationProfileReference.pdf
[AConfig] Apple Configurator 2 Help (online guidance)	http://help.apple.com/configurator/mac/2.6//
[DEP_Guide] Apple Deployment Programs Device Enrollment Program Guide	https://www.apple.com/business/docs/DEP_Guide.pdf
[PM_Help] Profile Manager Help	https://help.apple.com/profilemanager/mac/5.4/
[IOS_LOGS] Profiles and Logs	https://developer.apple.com/bug-reporting/profiles-and-logs/?platforms=ios
[PASSCODE-Help] Use a passcode with your iPhone, iPad or iPod touch	https://support.apple.com/en-us/HT204060
App Developer Guidance	
[3CC-MAN] Common Crypto man pages	https://developer.apple.com/legacy/library/documentation/Darwin/Reference/ManPages/#
[CKTSREF] Certificate, Key, and Trust Services	https://developer.apple.com/documentation/security/certificate_key_and_trust_services
[CRYPTOGUIDE] Cryptographic Services Guide	https://developer.apple.com/library/mac/documentation/Security/Conceptual/cryptoservices/Introduction/Introduction.html
[iOS_MDM] Mobile Device Management Protocol Reference	https://developer.apple.com/library/content/documentation/Miscellaneous/Reference/MobileDeviceManagementProtocolRef/1-Introduction/Introduction.html

Document Name	Location
[IPLKEYREF] Information Property List Key Reference	https://developer.apple.com/library/ios/documentation/General/Reference/InfoPlistKeyReference/Introduction/Introduction.html
[KEYCHAINPG] Keychain Services Programming Guide	https://developer.apple.com/library/ios/documentation/Security/Conceptual/keychainServConcepts/01introduction/introduction.html
[SECFWREF] Secure Framework	https://developer.apple.com/library/prerelease/ios/documentation/Security/Reference/SecurityFrameworkReference/index.html
[HTTPSTN2232] Technical Note TN 2232: HTTPS Server Trust Evaluation	https://developer.apple.com/library/ios/technotes/tn2232/_index.html

Table 1: TOE Guidance Documents

1.2 Evaluated TOE Configuration

Table *2*: Devices Covered by the Evaluation, lists the iPhone and iPad devices that are covered by the CC evaluation.

Device Name	Model Number	Processor
iPhone 5s	A1533 (GSM) A1533 (CDMA) A1453 A1457 A1530	A7
iPhone 6 Plus/ iPhone 6	A1549/A1522 (GSM) A1549/A1522 (CDMA) A1586/A1524	A8
iPhone 6s Plus iPhone 6s	A1634/A1633 (US) A1687/A1688 (Global)	A9
iPhone 7 Plus/ iPhone 7	A1784/A1778 (GSM) A1661/A1660 (CDMA)	A10 Fusion
iPhone 8 Plus iPhone 8	A1864/A1898/A1899/A1897 A1863/A1906/A1907/A1905	A11 Bionic
iPhone X	A1865/A1902 A1901	A11 Bionic
iPhone SE	A1662 (US) A1723 (Global)	A9
iPad mini 3	A1599 (Wi-Fi only) A1600 (Wi-Fi + cellular) A1601 (Wi-Fi + cellular)	A7
iPad mini 4	A1538 (Wi-Fi only) A1550 (Wi-Fi + cellular)	A8
iPad Air 2	A1566 (Wi-Fi only) A1567 (Wi-Fi + cellular)	A8X
iPad Pro 12.9”	A1584 (Wi-Fi only) A1652 (Wi-Fi + cellular)	A9X
iPad Pro 9.7”	A1673 (Wi-Fi only) A1674 (Wi-Fi + cellular)	A9X

Device Name	Model Number	Processor
iPad	A1822 (Wi-Fi only) A1823 (Wi-Fi + cellular)	A9
iPad Pro 12.9”	A1670 (Wi-Fi) A1671 (Wi-Fi + Cellular)	A10X Fusion
iPad Pro 10.5”	A1701 (Wi-Fi) A1709 (Wi-Fi + Cellular)	A10X Fusion

Table 2: Devices Covered by the Evaluation

1.3 Assumptions

The following assumptions apply when operating the TOE in the evaluated configuration. These assumptions must be complied with by the organization through the implementation of appropriate organizational policies and procedures.

1.3.1 [PP_MD_V3.1] Assumptions

- TOE administrators will configure the mobile device's security functions correctly to create the intended security policy.
- The mobile device user will immediately notify the administrator if the mobile device is lost or stolen.
- The mobile device user exercises precautions to reduce the risk of loss or theft of the mobile device.

1.3.2 [PP_WLAN_CLI_EP_V1.0] Assumptions

- Information cannot flow between the wireless client and the internal wired network without passing through the TOE.
- TOE Administrators are trusted to follow and apply all administrator guidance in a trusted manner.

1.3.3 [EP_MDM_AGENT_V3.0] Assumptions

- The TOE relies on network connectivity to carry out its management activities. The TOE will robustly handle instances when connectivity is unavailable or unreliable.
- The MDM Agent relies upon mobile platform and hardware evaluated against the [PP_MD_V3.1] and assured to provide policy enforcement as well as cryptographic services and data protection. The mobile platform provides trusted updates and software integrity verification of the MDM agent.
- One or more competent, trusted personnel who are not careless, willfully negligent, or hostile, are assigned and authorized as the TOE Administrators, and do so using and abiding by guidance documentation.
- Mobile device users are not willfully negligent or hostile and use the device within compliance of a reasonable Enterprise security policy.

1.4 TOE Security Functionality (TSF)

In the evaluated configuration, the TOE provides the following security functionality required by [PP_MD_V3.1], [EP_MDM_AGENT_V3.0], and [PP_WLAN_CLI_EP_V1.0].

- Security Audit
- Cryptographic support
- User data protection
- Identification and authentication
- Security management
- Protection of the TOE Security Functionality (TSF)
- TOE access
- Trusted path/channels

2 Secure Installation and Delivery

2.1 Secure Installation and Delivery of the TOE

The evaluated devices (TOE devices) are intended for end users who are employees of entities such as business organizations and government agencies.

The administrator of the customer entity is responsible for performing the necessary configuration to ensure that the TOE devices are placed in a configuration specified by the Security Target (ST).

The guidance documentation referenced in this CC Guide for the TOE devices can be accessed and downloaded from the Apple website as given in Table *1*: TOE Guidance Documents.

The normal distribution channels for obtaining these devices include the following:

- The Apple Store (either a physical store or online at https://apple.com)
- Apple retailers
- Service carriers (e.g., AT&T, Verizon)
- Resellers

Business

There is a distinct online store for Business customers with a link from the “Apple Store.” From the link to the “Apple Store” (https://www.apple.com), go to the upper left of the page and click “Business Store Home.” Or, optionally, use the following link.

https://www.apple.com/us_smb_78313/shop

Government

Government customers can use the following link.

https://www.apple.com/r/store/government/

Additional

Large customers can also have their own Apple Store Catalog for their employees to purchase devices directly from Apple under their corporate employee purchase program.

2.2 Secure Software Updates

All iOS updates are digitally signed. The user can verify the software version of the TOE on the devices. Refer to section 3.7, *Obtain Version Information*, for more information.

Software updates to the TOE are released regularly to address emerging security concerns and also provide new features; these updates are provided for all supported devices simultaneously. Users receive iOS update notifications on the device and through iTunes. Updates are delivered wirelessly, encouraging rapid adoption of the latest security fixes.

The device startup process helps ensure that only Apple-signed code can be installed on a device. To prevent devices from being downgraded to older versions that lack the latest security updates, iOS uses a process called System Software Authorization. If downgrades were possible, an attacker who gains possession of a device could install an older version of iOS and exploit a vulnerability that has been fixed in the newer version.

On a device with an A7 or later A-series system on a chip (SoC) the Secure Enclave processor (SEP) also utilizes System Software Authorization to ensure the integrity of its software and prevent downgrade installations.

iOS software updates can be installed using iTunes or over-the-air (OTA) on the device. With iTunes, a full copy of iOS is downloaded and installed. OTA software updates download only the components required to complete an update, rather than downloading the entire OS, improving network efficiency. Additionally, software updates can be cached on a local network server running the caching service on OS X Server so that iOS devices do not need to access Apple servers to obtain the necessary update data.

During an iOS upgrade, iTunes (or the device itself, in the case of OTA software updates) connects to the Apple installation authorization server and sends it a list of cryptographic measurements for each part of the installation bundle to be installed (for example, low-level bootloader (LLB), iBoot, the kernel, and OS image), a random anti-replay value (nonce), and the device's unique Electronic Chip ID (ECID).

The authorization server checks the presented list of measurements against versions for which installation is permitted and, if it finds a match, adds the ECID to the measurement and signs the result. The server passes a complete set of signed data to the device as part of the upgrade process. Adding the ECID "personalizes" the authorization for the requesting device. By authorizing and signing only for known measurements, the server ensures that the update takes place exactly as provided by Apple.

The boot-time chain-of-trust evaluation verifies that the signature comes from Apple and that the measurement of the item loaded from disk, combined with the device's ECID, matches what was covered by the signature.

These mechanisms ensure that the authorization is for a specific device and that an old iOS version from one device cannot be copied to another. The nonce prevents an attacker from saving the server's response and using it to tamper with a device or otherwise alter the system software.

Note that this ensures the integrity and authenticity of software updates. A TLS trusted channel is provided for this process.

2.3 Unevaluated Functionalities

The following security functionalities were not evaluated as part of the iOS 11.2 TOE and considered outside the scope of the evaluation.

2.3.1 Two-Factor Authentication

According to the [iPad_UG] and the [iPhone_UG] Privacy and security, Security, Two-factor authentication, Two-factor authentication is an extra layer of security for your Apple ID designed to ensure that you're the only person who can access your account, even if someone knows your password. It's built into iOS 9 and later, and OS X 10.11 and later.

This feature is outside the scope of the evaluated configuration.

2.3.2 Bonjour

According to the [iOSDeployRef], Bonjour is Apple's standards-based, zero configuration network protocol that lets devices find services on a network.

This feature is outside the scope of the evaluated configuration.

2.3.3 Unsupported VPN Protocols and Authentication Methods

The use of the following Virtual Private Network (VPN) protocols (and their authentication methods) which are described in the [iOSDeployRef] are outside the scope of the evaluated configuration.

- Cisco IPSec

- Layer Two Tunneling Protocol (L2TP) over IPSec
- Point-to-Point Tunneling Protocol (PPTP)

In addition, the following authentication methods are unsupported.

- L2TP over IPSec: user authentication by Microsoft Challenge Handshake Authentication Protocol version 2 (MS-CHAP v2) password, two-factor token, machine authentication by shared secret
- Cisco IPSec: user authentication by password, two-factor token, machine authentication by shared secret and certificates
- PPTP: user authentication by MS-CHAP v2 password, two-factor token
- Secure Sockets Layer (SSL) VPN: user authentication by password, two-factor token, certificates

2.3.4 VPN Split Tunnel

VPN split tunnel as described in the [iOSDeployRef] is outside of the evaluated configuration.

2.3.5 Siri Interface

The Siri interface as described in the [iPad_UG] and [iPhone_UG]

Since the Siri interface supports some commands related to configuration settings (For example, switching WiFi and Bluetooth on and off) the Siri interface must be turned off.

To turn Hey Siri on or off, go to *Settings»Siri & Search»Listen for “Hey Siri”*.

Siri can also be disabled using a configuration profile setting as described in Section 3.1, *Configuration Profiles*.

3 Administrative Guidance

This section provides additional guidance to configure and operate the evaluated configuration of the TOE.

Table 3: SFR Configuration Requirements, identifies whether each security functional requirement (SFR) requires any configuration to put the TOE in the evaluated configuration. For those SFRs that require configuration, the instructions are provided in the remainder of this document.

SFR	Function Description	Configurable?	Configuration Profile Needed?	Provided Guidance
FAU_ALT_EXT.2 (AGENT)	MDM agent provides alerts to MDM server via trusted channel	No	Yes	Section 3.2.13
FAU_GEN.1(1) (MDF)	Audit data generation	Yes	Yes	Section 3.6.1
FAU_GEN.1(2) (AGENT)	Audit data generation (Agent)	Yes	Yes	Section 3.6.1
FAU_SEL.1(2) (AGENT)	Audit event selection (Agent)	Yes	Yes	Section 3.6.1
FAU_STG.1 (MDF)	Audit storage protection	No—Audit records are not accessible to TOE Admins or Users and must be viewed on a trusted workstation or MDM server.	No	N/A
FAU_STG.4 (MDF)	Audit data loss prevention	Yes	Yes	Section 3.6.1
FCS_CKM.1(1) (MDF)	Rivest-Shamir-Adleman (RSA) and elliptic curve cryptography (ECC) key generation	No—the application programming interface (API) allows specification of the requested key sizes and key types.	No	Section 3.2.1
FCS_CKM.1(2) (WLAN)	Generation of symmetric keys for wireless LAN (WLAN)	No—WLAN keys are generated for the cipher suite offered by the access point.	No	N/A
FCS_CKM.2(1) (MDF)	RSA and ECC based key establishment	No—the API allows specification of the requested key sizes and key types.	No	Section 3.2.2
FCS_CKM.2(2) (MDF)	RSA and ECC based key establishment (Locked)	No—key establishment is hard coded.	No	N/A

SFR	Function Description	Configurable?	Configuration Profile Needed?	Provided Guidance
FCS_CKM.2(3) (WLAN)	Key distribution (for decryption of DEK)	No—the WLAN protocol is implemented according to IEEE 802.11 2012.	No	N/A
FCS_CKM_EXT.1 (MDF)	Root Encryption Key (REK)	No—REK is permanently etched in silicon and is both unmodifiable as well as inaccessible by iOS and apps.	No	N/A
FCS_CKM_EXT.2 (MDF)	Data Encryption Key (DEK)—key random generation	No—generation and maintenance of DEK is hard coded.	No	N/A
FCS_CKM_EXT.3 (MDF)	Key Encryption Key (KEK) generation	No—generation and maintenance of KEK is hard coded.	No	N/A
FCS_CKM_EXT.4 (MDF)	Zeroization of keys	No—zeroization of keys is hard coded.	No	N/A
FCS_CKM_EXT.5 (MDF)	Wiping of TSF data	Yes	No	Section 3.2.5
FCS_CKM_EXT.6 (MDF)	Salt generation	No—generation and maintenance of Salt is hard coded.	No	N/A
FCS_CKM_EXT.7 (MDF)	Root Encryption Key (REK)	No—REK is permanently etched in silicon and is both unmodifiable as well as inaccessible by iOS and apps.	No	N/A
FCS_COP.1(1) (MDF)	Advanced Encryption Standard (AES) encryption/decryption	No—for AES operations performed by the TSF. No—for AES operations performed by third party where the API allows specification of the AES cipher type	No	N/A for operations performed by TSF Section 3.2 for API

SFR	Function Description	Configurable?	Configuration Profile Needed?	Provided Guidance
FCS_COP.1(2) (MDF)	Secure Hash Algorithm (SHA)-1 and SHA-2 hashing	No—for hash operations performed by the TSF. No—for hash operations performed by third party where the API allows specification of the hash cipher type	No	N/A for operations performed by TSF Section 3.2.3 for API
FCS_COP.1(3) (MDF)	RSA and Elliptic Curve Digital Signature Algorithm (ECDSA) signature generation/verification	No—for signature operations performed by TSF No—for signature operations performed by third party where the API allows specification of the hash cipher type	No	N/A for operations performed by TSF Section 3.2 for API
FCS_COP.1(4) (MDF)	Keyed-hash message authentication code (HMAC) hashing	No—for HMAC operations performed by TSF No—for HMAC operations performed by third party where the API allows specification of the hash cipher type	No	N/A for operations performed by TSF Section 3.2.3 for API
FCS_COP.1(5) (MDF)	Password-based key derivation (PBKDF)	No—generation and maintenance of PBKDF is hard coded.	No	N/A
FCS_HTTPS_EXT.1 (MDF)	HTTPS protocol	No—the used HTTPS cipher suite is defined by the HTTPS server where all cipher suites listed in the ST are always available.	No	N/A
FCS_IV_EXT.1 (MDF)	Initialization vector (IV) generation	No—generation and maintenance of IVs is hard coded.	No	N/A
FCS_RBG_EXT.1 (MDF)	Random bit generation	No—generation of random numbers is hard coded.	No	N/A
FCS_SRV_EXT.1 (MDF)	Cryptographic algorithm services	See all statements for FCS_COP.1, FCS_CKM.1, and FCS_CKM.2 above.	See above.	See above.

SFR	Function Description	Configurable?	Configuration Profile Needed?	Provided Guidance
FCS_STG_EXT.1 (MDF)	Key storage	No—the API allows specification of the requested key sizes and key types.	No	Section 3.2.6
FCS_STG_EXT.2 (MDF)	DEK and KEK encryption	No—generation and maintenance of DEK and KEK is hard coded.	No	N/A
FCS_STG_EXT.3 (MDF)	DEK and KEY protection	No—generation and maintenance of DEK and KEK is hard coded.	No	N/A
FCS_STG_EXT.4 (AGENT)	MDM agent protected storage of private keys and secrets by the MDM	No	No	N/A
FCS_TLSC_EXT.1 (MDF)	Extensible Authentication Protocol-Transport Layer Security (EAP-TLS) protocol	Yes	Yes	Section 3.2.7
FCS_TLSC_EXT.1/ WLAN (WLAN)	TLS protocol	No—used TLS cipher suites are defined by the TLS server where all cipher suites listed in the ST are always available. The API of the third-party application defines specific TLS protocol rules.	No	Section 3.2.8
FCS_TLSC_EXT.2	EAP TLS and TLS	Yes	Yes	Section 3.2.7
FDP_ACF_EXT.1 (MDF)	Security access control	No—access control settings are hard coded.	No	N/A

SFR	Function Description	Configurable?	Configuration Profile Needed?	Provided Guidance
FDP_DAR_EXT.1 (MDF)	Protected data encryption	No—data is always encrypted. TSF is hard coded to use the appropriate data protection levels. Third party applications, however, can choose the data protection level with API calls. By default, all files created within third party apps are protected as "Class C," but they can request via the APIs to elevate or demote that level of protection.	No	Section 3.3.1
FDP_DAR_EXT.2 (MDF)	Sensitive data encryption	No—data is always encrypted. TSF is hard coded to use the appropriate data protection level. Third party applications, however, can choose the data protection level with API calls. By default, all files created within third party apps are protected as "Class C," but they can request via the APIs to elevate or demote that level of protection.	No	Section 3.3.1
FDP_IFC_EXT.1 (MDF)	VPN subset information flow control	Yes	Yes	Section 3.1
FDP_PBA_EXT.1 (MDF)	Biometric Authentication Factors	Yes—Use of BAFs can be enabled or disabled.	Yes	Section 3.4.1

SFR	Function Description	Configurable?	Configuration Profile Needed?	Provided Guidance
FDP_STG_EXT.1 (MDF)	User data storage	No—the trust anchor database maintenance is hard coded. The administrator can add/remove their own Anchors of Trust to/from that database.	No	N/A
FDP_UPC_EXT.1 (MDF)	Inter-TSF user data transfer protection	For HTTPS and TLS, see above. No—for Bluetooth as its maintenance is hard coded	For HTTPS and TLS, see above. No—for Bluetooth	For HTTPS and TLS, see above. N/A for Bluetooth
FIA_AFL_EXT.1 (MDF)	Authentication failure	Yes	Yes	Section 3.4.3
FIA_BLT_EXT.1 (MDF)	Bluetooth user authorization	No—the Bluetooth protocol allows different types of authorization which are supported by the TOE. The used authorization type depends on the remote device capability.	No	Section 3.4.4
FIA_BLT_EXT.2 (MDF)	Bluetooth mutual authentication	No—Bluetooth mutual authentication is required prior to data transfer.	No	N/A
FIA_BLT_EXT.3 (MDF)	Rejection of duplicate Bluetooth connections	No—No device can establish duplicative Bluetooth connections.	No	N/A
FIA_BLT_EXT.4 (MDF)	Secure simple pairing	No—secure simple pairing cannot be disabled.	No	N/A
FIA_BMG_EXT.1 (MDF)	Biometric authentication	No	No	N/A
FIA_BMG_EXT.2 (MDF)	Biometric sample quality	No	No	N/A
FIA_BMG_EXT.3 (MDF)	Biometric sample quality	No	No	N/A
FIA_BMG_EXT.5 (MDF)	Biometric sample quality	No	No	N/A
FIA_ENR_EXT.2 (AGENT)	Device enrollment in management	Yes	Yes	Sections 3.4.8 & 3.5.10

SFR	Function Description	Configurable?	Configuration Profile Needed?	Provided Guidance
FIA_PAE_EXT.1 (WLAN)	Port Access Entity (PAE) authentication	No—the WLAN protocol is implemented according to IEEE 802.11 2012.	No	N/A
FIA_PMG_EXT.1 (MDF)	Password management	Yes	Yes	Section 3.4.1
FIA_TRT_EXT.1 (MDF)	Authentication throttling	No—the authentication delay is hard coded.	No	N/A
FIA_UAU.5 (MDF)	Multiple authentication mechanisms	Yes	No	Section 3.1
FIA_UAU.6(1) (MDF)	Re-authentication	No—users must be re-authenticated before any changes to the password authentication factor can be made.	No	N/A
FIA_UAU.6(2) (MDF)	Re-authentication (Locked)	No	No	Section 3.4.5
FIA_UAU.7 (MDF)	Protected authentication feedback	No (enabled by default)	No	Section 3.4.5
FIA_UAU_EXT.1 (MDF)	Authentication for cryptographic operation	Yes—the user must set a passphrase to enable authentication token protection.	Yes—by specifying a passphrase quality rule which mandates the setting of a passphrase	Section 3.4.1
FIA_UAU_EXT.2 (MDF)	Timing of authentication	Yes	Configurable with UI as well as profile	Section 3.5.2
FIA_X509_EXT.1 (MDF)	Certificate validation	No—the certificate validation rules are hard coded.	No	N/A
FIA_X509_EXT.2 (MDF)	Certificate authentication	Yes—the certificates required for authentication must be provided.	Yes	Section 3.4.7
FIA_X509_EXT.2 (WLAN)	Certificate authentication (EAP-TLS)	Yes	Yes	Section 3.4.6

SFR	Function Description	Configurable?	Configuration Profile Needed?	Provided Guidance
FIA_X509_EXT.3 (MDF)	Certificate validation request	No—the API is provided with certificate validation rules hard coded.	No	Section 3.4.7.1
FMT_MOF_EXT.1 (MDF)	Security function behavior management	Yes	Yes	Section 3.5
FMT_POL_EXT.2 (AGENT)	Trusted policy update	No—	No	N/A
FMT_SMF_EXT.1 (MDF)	Management function specification	Yes	Yes	Section 3.5
FMT_SMF_EXT.1/ WLAN (WLAN)	Management function specification	Yes	Yes	Section 3.1
FMT_SMF_EXT.2 (MDF)	Remediation action specification	Yes	Yes	Section 3.5
FMT_SMF_EXT.3 (AGENT)	Management function specification (Agent)	Yes	Yes	Sections 3.4.8 & 3.5.11
FMT_UNR_EXT.1 (AGENT)	User unenrollment prevention	Yes	Yes	Section 3.5.11
FPT_AEX_EXT.1 (MDF)	Anti-exploitation services (address-space layout randomization)	No—the service is hard coded.	No	N/A
FPT_AEX_EXT.2 (MDF)	Anti-exploitation services (memory page permissions)	No—the service is hard coded.	No	N/A
FPT_AEX_EXT.3 (MDF)	Anti-exploitation services (overflow protection)	No—the service is hard coded.	No	N/A
FPT_AEX_EXT.4 (MDF)	Domain isolation	No— the service is hard coded.	No	N/A
FPT_JTA_EXT.1 (MDF)	Joint Test Action Group (JTAG) Disablement	No—JTAG interfaces are not present on iOS devices.	No	N/A
FPT_KST_EXT.1 (MDF)	Plaintext key storage	No—keys are stored in secure enclave or in key chain. Wrapped keys are stored in Effaceable Storage.	No	N/A
FPT_KST_EXT.2 (MDF)	No key transmission	No—keys are stored in secure enclave or in key chain.	No	N/A

SFR	Function Description	Configurable?	Configuration Profile Needed?	Provided Guidance
FPT_KST_EXT.3 (MDF)	No plaintext key export	No—keys are stored in secure enclave that does not provide key export facility. The TOE does not export keys stored in key chain.	No	N/A
FPT_NOT_EXT.1 (MDF)	Self-test notification	No	No	N/A
FPT_STM.1 (MDF)	Reliable timestamps	Yes	No	Section 3.5.4
FPT_TST_EXT.1 (MDF)	TSF cryptographic functionality testing	No	No	N/A
FPT_TST_EXT.1/ WLAN (WLAN)	TSF cryptographic functionality testing (Wireless)	No	No	N/A
FPT_TST_EXT.2 (MDF)	TSF integrity testing	No	No	N/A
FPT_TST_EXT.3 (MDF)	TSF integrity testing	No	No	N/A
FPT_TUD_EXT.1 (MDF)	Trusted update (TSF version query)	No	No	N/A
FPT_TUD_EXT.2 (MDF)	Trusted update verification	No	No	N/A
FTA_SSL_EXT.1 (MDF)	Session locking	Yes	Configurable with UI as well as profile	Section 3.5.3
FTA_TAB.1 (MDF)	Banner	Yes	No	Section 3.5.5
FTA_WSE_EXT.1 (WLAN)	Wireless network access	Yes	Yes	Section 3.1
FTP_ITC_EXT.1(2) (AGENT)	Trusted channel communication (TLS, HTTPS, IPSec)	See FCS_HTTPS_EXT.1, FCS_TLSC_EXT.1, FCS_TLSC_EXT.2 and FDP_IFC_EXT.1.	See FCS_HTTPS_EXT.1, FCS_TLSC_EXT.1, FCS_TLSC_EXT.2 and FDP_IFC_EXT.1.	See FCS_HTTPS_EXT.1, FCS_TLSC_EXT.1, FCS_TLSC_EXT.2 and FDP_IFC_EXT.1, section 3.2.11.

SFR	Function Description	Configurable?	Configuration Profile Needed?	Provided Guidance
FTP_ITC_EXT.1(3) (WLAN)	Trusted channel communication (EAP-TLS)	No	No	N/A

Table 3: SFR Configuration Requirements

3.1 Configuration Profiles

Many aspects of the claimed TOE security functionality are configured using “Configuration Profiles” that are installed on the TOE. Configuration Profiles are Extensible Markup Language (XML) files that may contain settings for a number of configurable parameters. For details on the different payloads and keys that can be defined, see the document “Configuration Profile Reference” [IOS_CFG], “Over-the-Air Profile Delivery and Configuration” [OTA_Guide] and “Mobile Device Management Protocol Reference [iOS_MDM] listed in Table *1*: TOE Guidance Documents, that can also be downloaded from the Apple website.

The Configuration Profiles can be deployed in one of the following five different ways.

- Using the Apple Configurator tool
- Via an email message
- Via a web page
- Using over-the-air configuration
- Using over-the-air configuration via a Mobile Device Management Server

The following mandatory configurations must be configured using Configuration Profiles.

- The passcode policy—the administrator can define this using the Passcode Policy Payload to:
    - define the minimum passcode length,
    - define requirements for the passcode complexity,
    - define the maximum passcode lifetime,
    - define the maximum time of inactivity after which the device is locked automatically, and
    - define the maximum number of consecutive authentication failures after which the device is wiped.
- The Passcode Policy Payload is provided in more detail in section 3.4.1, *Passcode Authentication Configuration*.
- The VPN policy can be defined as follows.
    - If VPN is always on, or if per-App VPN is used
    - Define the authentication method (Shared Secret or Certificate) (for IPSec Internet Key Exchange (IKE)v2 based VPN)
    - Specification of certificates or shared keys

VPN policies for per-App VPN are defined using the per-App VPN Payload, and VPN policies for AlwaysOn VPN or VPN on demand are defined using the VPN Payload where the VPNType key defines which of the two options are taken. To have AlwaysOn VPN, set the key to AlwaysOn, otherwise set the key to IKEv2 or IPSec and set the value of the key OnDemandEnabled to 1.

Note: VPN on demand is outside the scope of the evaluation. Internet Key Exchange Version 2 (IKEv2) can be configured using the IKEv2 Dictionary Keys, which requires the specification of the IP address or hostname of the VPN server, the specification of the client identifier, the specification of the remote identifier, the specification of the authentication method (shared secret or certificate), and the specification of either the shared secret or the certificate used for authentication. Optional keys allow the enablement of extended authentication, the specification of a username and password, the specification of the interval the connection is kept alive when the peer cannot be reached, the specification of the Common Name of the server certificate issuer and/or the Common Name of their server certificate, and the specification of the IKE Security Association Parameters and the child security association parameters. Those parameters include the specification of the encryption algorithm (values allowed for the evaluated configuration are Advanced Encryption Standard (AES)-128 and AES-256) note that the use of DES and 3DES is not allowed in the evaluated configuration, the specification of the hash function used for integrity verification (allowed values for the evaluated configuration are SHA1-160, SHA2-256, SHA2-384, and SHA2-512), and the specification of the Diffie-Hellman Group for IPSec (allowed values for the evaluated configuration are 14 through 21). When the option AlwaysOn is selected, in addition to the IKEv2 configuration (IKEv2 is the only option for the protocol in this case) the interfaces (Wi-Fi or Cellular or both) for which AlwaysOn VPN applies also need to be defined. Exceptions for AlwaysOn VPN for VoiceMail and/or AirPrint can be defined if an organization's policy allows those.

An example of settings for AlwaysOn VPN is provided in Table *4*: VPN Payload

Setting	**Description**
UserDefinedName	< Description of the VPN connection displayed on the device.>
OverwritePrimary	True
VPNType	AlwaysOn
OnDemandEnabled	0
IKEv2 Dictionary Keys	
Remote address	<IP address or hostname of your organization’s VPN server>
LocalIdentifier	<specify the format as allowed in iOS>
RemoteIdentifier	<specify the format as allowed in iOS>
AuthenticationMethod	Certificate
PayloadCertificateUUID	<The universally unique identifier (UUID) of the identity certificate>
ExtendedAuthEnabled	<1 if EAP is enabled, 0 otherwise.

Table 4: VPN Payload

- The Wi-Fi policy for connecting to dedicated Service Set Identifiers (SSIDs):
  - The Extensible Authentication Protocol (EAP) types allowed
  - The SSIDs allowed to connect to
  - The encryption type(s) allowed

Note: iOS does not support the general definition of a whitelist of SSIDs that a device can connect to. Such a whitelist can only be defined for VPN on demand, where such a whitelist can be defined as part of the On Demand Rules. VPN on demand, however, is outside the scope of the evaluation.

Other Configuration Profile restrictions:

  - Disallow the modification of wallpaper (key: allowWallpaperModification)
  - Disable the “Show Control Center in Lock screen” option
  - Disallow the Siri interface (key: allowAssistant)

- General restrictions defined by keys in the Restrictions Payload:
  - Allowing or disallowing specific services (e.g., remote backup) (key: AllowCloudBackup) or using the cameras (key:allowCamera)
  - Allowing or disallowing notifications when locked (key: allowLockScreenNotificationView)
  - Allowing or disallowing a prompt when an untrusted certificate is presented in a TLS/HTTPS connection (key: allowUntrustedTLSPrompt)

In addition, the following functions can be enabled/disabled by an administrator using Configuration Profiles.

- Installation of applications by a user (key: allowAppInstallation)
- Possibility to perform backups to iCloud (key: allowCloudBackup)
- Ability to submit diagnostics automatically (key: allowDiagnosticSubmission)
- Ability to use the fingerprint device (Touch ID) for user authentication (key: allowFingerprintForUnlock)
- Ability to see notifications on the lock screen (key: allowLockScreenNotificationsView)
- Ability to take screen shots (key: allowScreenShot)
- Ability to accept untrusted TLS certificates (key: allowUntrustedTLSPrompt). *Note: This is part of the evaluated configuration to the extent that an administrator has the option to enable it, however, there is no requirement to enable it.*
- Ability to perform unencrypted backups (via iTunes) (key: forceEncryptedBackup)

Further additional restrictions can be enforced for enrolled devices using Configuration Profiles including the following.

- Disallow the use of AirDrop
- Disallow the modification of the account or the change to cellular data usage
- Disallow the use of specific services like the iBookstore, the Messages app, Game Center, USB pairing with a host different from the supervision host, user installation of configuration profiles, use of podcasts, definition lookup
- Disable the “Erase All Content And Settings" option in the Reset User Interface
- Disable the enablement of restrictions in the Restrictions User Interface
- Definition of a whitelist for AirPlay
- Locking the device to a single application
- Specification of a global HTTP proxy
- Specification of URL whitelists and blacklists
- Disable the “allow pairing with non-Configurator hosts” option
- Disable the “Allow Passbook notifications in Lock screen” option
- Disable the “Show Notification Center in the Lock screen” option
- Disable the “Show Today view in Lock screen” option
- Disable the “Allow Siri while device is locked” option

A user can access management functions available to him via the menus of the graphical user interface. The functions the user can perform are described in the [iPhone_UG] and the [iPad_UG]. More information is provided in the remainder of this document.

3.2 Crypto-Related Function Configuration

The TOE comes with the following two cryptographic modules that provide the cryptographic support for the TOE.

- Apple CoreCrypto Cryptographic Module v8 for ARM

- Apple CoreCrypto Kernel Cryptographic Module v8 for ARM

Warning: Use of any other cryptographic module that was not evaluated or tested during CC evaluation of the TOE will take the operating environment out of the evaluated configuration.

API Documentation for cryptographic functions performed by the TOE is provided in the documents: “Cryptographic Services Guide” [CRYPTOGUIDE] and “Certificate, Key, and Trust Services” [CKTSREF] which cover the following functions.

- SecKeyCreateEncryptedData for encryption
- SecKeyCreateDecryptedData for decryption
- SecKeyCreateSignature for signature generation
- SecKeyVerify Signature for signature verification

Symmetric cryptographic functions are provided by the Common Crypto API which is a thin wrapper layer to CoreCrypto. AES encryption and decryption are performed via the following function calls described in the Cryptographic Services Guide [CRYPTOGUIDE].

- CCCryptorCreate
- CCCryptorRelease
- CCCryptorUpdate
- CCCryptorFinal
- CCCrypt

This includes AES-CBC, AES-CCMP, AES-GCM, AES-CCM, and AES Key Wrap for 128-bit and 256-bit keys.

3.2.1 Key Generation, Signature Generation and Verification

The TOE generates the following asymmetric keys.

- RSA with key sizes of 2048 bits or greater
- ECC with NIST curves P-256 and P-384 with key sizes of 256 bits and 384 bits, respectively
- ECC curve 25519, with a key size of 256 bits

iOS provides a programming interface for key pair generation (function SecKeyGeneratePair, described in the “Certificate, Key, and Trust Services” [CKTSREF]). The 'kSecAttrKeyType' key passed in with the 'parameters' parameter defines the type of key pair and the 'kSecAttrKeySizeInBits' key defines the key size. The following are defined key types for asymmetric keys.

- kSecAttrKeyTypeRSA
- kSecAttrKeyTypeECSECPrimeRandom

For the evaluated configuration, key generation does not require additional configuration by the end user as the API allows specification of the requested key sizes and key types.

iOS provides programming interfaces for signature generation (function SecKeyRawSign) and verification (function SecKeyRawVerify) using RSA schemes and ECDSA schemes. These functions are described in the “Certificate, Key, and Trust Services” [CKTSREF].

3.2.2 Key Establishment

The TOE performs the following key establishment.

- RSA-based scheme
- ECC-based scheme

Key establishment is performed for TLS and IKE. An application that uses the library functions defined in the *Secure Transport Reference* section of the “Security Framework Reference” [SECFWREF] can obtain the cipher suites iOS supports using the function 'SSLGetSupportedCiphers' and limit the cipher suites offered in the client hello message using the function 'SSLSetEnabledCiphers'.

For the evaluated configuration, no configuration is required from the end user.

3.2.3 Hashing

The TOE performs the following hash functions: SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 with message digest sizes 160, 256, 384, and 512 bits.

Functions to perform hashing are provided as part of the Common Crypto library. The functions that perform cryptographic hashing are as follows and are described in the Cryptographic Services Guide [CRYPTOGUIDE] and in detail in the Common Crypto man pages [3CC-MAN].

- CC_SHA1_Init, CC_SHA1_Update, CC_SHA1_Final, and CC_SHA1 for SHA-1
- CC_SHA256_Init, CC_SHA256_Update, CC_SHA256_Final, and CC_SHA256 for SHA-256
- CC_SHA384_Init, CC_SHA384_Update, CC_SHA384_Final, and CC_SHA384 for SHA-384
- CC_SHA512_Init, CC_SHA512_Update, CC_SHA512_Final, and CC_SHA512 for SHA-512

All these functions require the length of string to be hashed to be defined in bytes.

The choice of SHA function used is dictated by the invoking function, the user has no capability to configure this choice. Each TLS ciphersuite uses a specific and appropriate SHA function and here again, the user does not have the ability to affect the choice through configuration.

The following functions to perform keyed hashing are also provided and are described in the Cryptographic Services Guide [CRYPTOGUIDE] and in detail in the Common Crypto man pages [3CC-MAN].

- CCHmacInit
- CCHmacUpdate
- CCHmacFinal
- CCHmac

These functions call the appropriate HMAC algorithm for the size of the specified digest. These include the following HMAC functions.

- HMAC-SHA-1
- HMAC-SHA-224
- HMAC-SHA-256
- HMAC-SHA-384
- HMAC-SHA-512 with message digest sizes 160,256, 384, and 512 bits

For the evaluated configuration, no configuration is required from the end user.

3.2.4 Random Number Generation

For the evaluated configuration, no additional configuration is required from the end user.

3.2.5 Wiping of Protected Data

A wipe operation is performed after the user exceeds the limit number of failed authentication attempts or upon receiving a request from an authorized administrator. Wiping is performed by clearing the block with the highest level of KEKs using a block erase of the 'effaceable area' of the non-volatile flash memory.

The administrator can issue a remote wipe command from the MDM server.

The user can set the TOE device to erase all data after 10 failed passcode attempts. To do this, go to *Settings»Touch ID & Passcode* (TOE models with Touch ID) or *Settings»Face ID & Passcode* (iPhone X), and enable ‘Erase Data’.

3.2.6 Keys/Secrets Import/Destruction

Cryptographic keys are stored in keychains. The API documentation for management of keys/secrets (i.e., import, use, destroy) is provided in the *Managing Keys, Certificates, and Passwords* section of the “Cryptographic Services Guide” [CRYPTOGUIDE], with the API specified in the “Certificate, Key, and Trust Services” [CKTSREF] and the “Keychain Services Programming Guide” [KEYCHAINPG], which describes how keychain items are created, managed, and deleted. There it is described that in iOS an application has only access to its own keychain items, so access restrictions are automatically satisfied.

3.2.7 EAP-TLS Configuration

For EAP-TLS, the TOE implements TLS 1.0, TLS 1.1 and TLS 1.2 supporting the cipher suites listed in Table *5*: EAP-TLS Ciphersuites, below. [PP_MD_V3.1] limits the cipher suites used by EAP-TLS connections in the evaluated configuration.

In the evaluated configuration, the TOE must use only the EAP-TLS cipher suites listed in Table *5*: EAP-TLS Ciphersuites.

Ciphersuite Name
TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256

Table 5: EAP-TLS Ciphersuites

The cipher suites in Table *5*: EAP-TLS Ciphersuites above, are automatically selected by the TOE (i.e., the TOE does not support the individual selection of EAP-TLS cipher suites) when Wi-Fi Protected Access (WPA)-EAP is configured as follows.

WI-FI:

- SSID: <name of SSID>
- No hidden network
- AutoJoin
- No Proxy
- Security Type: WPA2 Enterprise (iOS 8 or later)

- Accepted EAP Types: TLS
- Identity certificate: <select certificate of to be used by the client>
- Network Type: Standard
- Trust tab: <mark the CA certificate as trusted>
- Trust: Add the name of the access point / EAP certificate to the list of Trusted Server Certificate Names

No additional configuration is needed for the automatic recovery of an unintentionally broken connection.

3.2.8 TLS Configuration

TLS is provided by the APIs of the iOS Security Framework, which uses the Apple CoreCrypto Cryptographic Module v8 for ARM.

The library implements TLS 1.2 supporting the cipher suites listed in Table *6*: TLS Ciphersuites. The [ST] limits the cipher suites used by TLS connections in the evaluated configuration.

The supported cipher suites below are automatically selected by the TOE (i.e., the TOE does not support the individual selection of TLS cipher suites). The TLS cipher suites available are defined by TLS server where all cipher suites listed in the ST are always available. Thus, no additional configuration is required by the end user.

When connected via WiFi, EAP-TLS is configured using a configuration profile, using the WLAN payload.

Ciphersuite Name
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

Table 6: TLS Ciphersuites

3.2.8.1 Setting Up TLS/HTTPS Channel

Guidance for setting up a TLS/HTTPS channel is provided in the following documents.

- “Security Framework Reference” [SECFWREF] in the chapter *Secure Transport Reference*
- “Technical Note TN 2232: HTTPS Server Trust Evaluation” [HTTPSTN2232] for additional HTTPS guidance

3.2.8.2 Setting Reference Identifier

Guidance/API documentation for setting the reference identifier for certification validation in TLS is provided in the Obtain policies for establishing trust section in the chapter Policies described in the [CKTSREF].

3.2.9 Certificate Authority (CA) Configuration

The iOS Trust Store contains trusted root certificates that are preinstalled with iOS. A list of the trusted certificates can be found at https://support.apple.com/en-us/HT204132.

Additional CAs can be added via the Apple Configurator. This can also be done using a Configuration Profile. See description for the keys for EAPClientConfiguration, PayloadCertificateArchorUUID and TLSTrustedServerNames in the [IOS_CFG]. There are also other keys that configure additional parameters of EAP-TLS.

3.2.10 Client Certificate Configuration

A client certificate with its keys can be installed on the device using a Configuration Profile. See section 3.1, *Configuration Profiles*, of this document for more details about Configuration Profiles.

3.2.11 Configuration of the Supported Elliptic Curves Extension

The supported elliptic curves below are automatically selected by the TOE (i.e., the TOE does not support the individual selection of elliptic curves). The [ST] limits the curves used by TLS connections in the evaluated configuration. The curves available are defined by the server where all curves listed in the ST are always available. Thus, no additional configuration is required by the end user.

Available curves: secp256r1, secp384r1.

x25519 is also supported by the TOE.

3.2.12 Configure MDM Agent and MDM Communications

MDM Agent-Server communication is achieved securely using the MDM protocol which is built on top of HTTP, TLS, and push notifications that use HTTP PUT over TLS (SSL). A managed mobile device uses an identity to authenticate itself to the MDM server over TLS (SSL). This identity can be included in the profile as a Certificate payload or can be generated by enrolling the device with Simple Certificate Enrollment Protocol (SCEP).

The MDM Agent communications uses the iOS Security Framework as described in section 3.2.8, *TLS Configuration*. Thus, configuring the TOE’s TLS protocol as per section 3.2.8 automatically configures the MDM Agent communications. If an additional CA certificate needs to be added to support the MDM Server, see section 3.2.9, *Certificate Authority (*CA) Configuration.

3.2.13 MDM Agent Alerts

The MDM Agent generates and sends an alert in response to an MDM server request for applying a configuration policy and in response to receiving a reachability event. These responses are always enabled.

When the application of a configuration policy to a mobile device is successful, the MDM Agent replies with an MDM Result Payload with Status value "Acknowledged".

When the application of a configuration policy is unsuccessful, the MDM Agent replies with an MDM Result Payload with Status value "Error" or CommandFormatError, "Idle" and "NotNow".

When a reachability event is received by the MDM Agent, the MDM Agent replies with an MDM Result Payload to acknowledge that the device received the event.

3.3 Data Protection Configuration

3.3.1 Data-At-Rest (DAR) Protection Configuration

Data is always encrypted for protection. However, to ensure data protection, establishment of a passcode on the device is required. The effectiveness of data protection depends on a strong passcode, which is required for the evaluated configuration.

Users can check that data protection is enabled on their device by looking at the passcode settings screen and also by seeing that a passcode is required to access the device. No further configuration is required.

3.3.2 VPN/Wi-Fi Configuration

VPN connections can be configured across a device or on a per-app basis, by configuring AlwaysOn VPN. The ‘AlwaysOn’ VPN configuration enables the organization to have full control over managed and supervised device traffic by tunneling all IP traffic back to the organization. Configuration of VPN/Wi-Fi is performed by an administrator using a Configuration Profile described in the [IOS_CFG]. See section 3.1, *Configuration Profiles*, for more details on Configuration Profiles and sample of the VPN Payload.

3.3.3 Restrict Application Access to System Services

Access control to system services is hardcoded thus not configured by the end user.

However, in order to restrict an application from accessing system services, an app has to declare the system services it wants to use in the Info.Plist file described in 'Information Property List Key Reference' [IPLKEYREF], chapter iOS Keys.

3.4 Identification & Authentication Configuration

3.4.1 Passcode Authentication Configuration

The Passcode Policy Payload is described in [iOS-CFG].

In the evaluated configuration, users must enable the use of a password/passcode. The TOE enforces the following parameters for passcode authentication.

- Passcodes shall be composed of any combination of upper and lower case letters, numbers, and special characters: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”
- Passcode length shall be up to 16 characters

Passcode policy is defined by the administrator using the Configuration Profile. The administrator can define this Configuration Profile using the Passcode Policy Payload described in the [IOS_CFG]. The Passcode Policy Payload presents the user with an alphanumeric passcode entry mechanism, which allows for the entry of arbitrarily long and complex passcodes including the selection of special characters. Set the configuration keys allowSimple to false and RequireAlphanumberic to Yes.

Also, set the configuration key “minLength” to a value defined by the organization's policy. A value of 10 or more characters is recommended.

Table *7*: Passcode Policy Payload, provides details of the key values that must be set in order to meet the requirements of the [ST].

Key	Setting
allowSimple	Must be set to 'false'
forcePin	Must be set to 'true'
maxFailedAttempts	Any value between 2 and 10. 10 is the default
maxInactivity	Needs to be set to a value defined by the organization's policy. A value of less than 5 minutes is recommended.
maxPINAgeInDays	Needs to be set to a value defined by the organization's policy. A value of less than 90 days is recommended.

Key	Setting
minComplexChars	Needs to be set to a value defined by the organization's policy.
minLength	Needs to be set to a value defined by the organization's policy. A value of 10 or more characters is recommended.
requireAlphanumeric	Needs to be set to a value defined by the organization's policy.
pinHistory	Needs to be set to a value defined by the organization's policy. A value of 5 or more is recommended.
maxGracePeriod	Must be set to 0
allowFingerprintModification	Needs to be set to a value defined by the organization's policy.
changeAtNextAuth	Needs to be set to a value defined by the organization's policy.

Table 7: Passcode Policy Payload

The passcode is obscured by default. No additional configuration is required. Additionally, biometric authentication factors do no relay authentication entry information and are inherently obscured.

An Administrator can configure if Touch ID can be enabled by a user by setting the key AllowFingerprintForUnlock to false in a Configuration Profile using the Restrictions Payload.

3.4.2 Biometric Authentication Factors

Enrollment and management of biometric authentication factors and credentials is detailed in the [iPhone_UG] and [iPad_UG] respectively.

Enrollment for Touch ID is typically accomplished during initial device configuration but can also be performed using the *Settings»Touch ID & Passcode* menus. Multiple fingerprints may be enrolled, named, and deleted from this menu. In order to remove a specific finger, a user must tap the finger for removal followed by delete fingerprint. Users may place a finger on the Touch ID sensor to determine which biometric credential entry it is mapped to. Users may also disable Touch ID selectively for applications, or entirely, from the *Settings»Touch ID & Passcode* menu, by authenticating using their passcode and turning off one or more of the following corresponding options.

- Unlock
- Apple Pay
- ITunes & App Store

Enrollment for Face ID is typically accomplished during initial device configuration but can also be performed using the *Settings»Face ID & Passcode* menu by tapping the “Set up Face ID” option. Face ID does not support multiple users and only one individual per device may establish a Face ID credential by providing biometric samples for enrollment. Users may remove Face ID biometric samples from the *Settings»Face ID & Passcode* menu by tapping the “Reset Face ID” option. Users may also disable Face ID selectively for applications, or entirely, from the *Settings»Face ID & Passcode* menu by turning off one or more of the following corresponding options.

- Unlock
- Apple Pay

- iTunes & App Store
- Safari AutoFill

3.4.3 Authentication Attempt Configuration

To limit/configure the number of consecutive failed authentication attempts for the passcode; the administrator can use a Configuration Profile. See the Passcode Policy Payload in section 3.4.1, *Passcode Authentication Configuration*. The configuration key “maxFailedAttempts” is set to 10 by default.

Both Face ID and Touch ID allow up to five unsuccessful authentication attempts before passcode authentication is required. See Section 3.4.5 Protected Authentication Feedback Configuration for more details.

3.4.4 Bluetooth Configuration

See the [iPhone_UG] and the [iPad_UG] *iPhone and other devices* section *Connect Bluetooth devices* for instructions how to turn Bluetooth on and off and how to pair and unpair a Bluetooth device. Bluetooth can also be disassociated via the Control Center.

Manual authorization is implicitly configured since pairing can only occur when explicitly authorized through the *Settings»Bluetooth* interface. During the pairing time, another device (or the iOS) can send a pairing request. Commonly, a six-digit number is displayed on both sides which must be manually matched by a user, i.e. the PIN is shown and the user must accept it before the pairing completes. If one device does not support this automatic exchange of a PIN, a window for entering a manual PIN is shown. That PIN must match on both sides.

iOS requires that remote Bluetooth devices support an encrypted connection.

Devices that want to pair with the TOE via Bluetooth are required by Apple to use Secure Simple Pairing, which uses Elliptic Curve Diffie-Hellman (ECDH) based authentication and key exchange. See the “Specification of the *Bluetooth* System,” 4.0, Volume 2, Part H, chapter 7 [BT] for details.

The only time the device is Bluetooth discoverable is when the Bluetooth configuration panel is active and in the foreground (there is no toggle switch for discoverable or not discoverable—unless the configuration panel is the active panel, the device is not discoverable).

Connections via Bluetooth/LE are secured using AES-128 in CCM mode. For details of the security of Bluetooth/LE see the “Specification of the *Bluetooth* System,” 4.0, Volume 6, Part E, chapter 1 [BT].

3.4.5 Protected Authentication Feedback Configuration

In the evaluated configuration, the passcode on the TOE devices is obscured by default. All passcode entries are obscured by dot symbol for each character as the user input occurs. Biometric authentication inputs do not produce feedback to the user unless an input is rejected. When an invalid fingerprint sample is given or cannot be authenticated, a simple error message is returned to the user to try again. When an invalid facial sample is given or cannot be authenticated, the device will vibrate. If three invalid biometric samples are presented the device will offer passcode entry. After five invalid biometric samples are presented passcode authentication is required. No configuration is needed. Refer to [PASSCODE_Help] for more information on how to manage a passcode.

3.4.6 Re-Authentication Configuration

When the use of a passcode is enabled, the device automatically prompts the user for a passcode to unlock the device. No additional configuration is required.

Use of Touch ID or Face ID can be set in the *Settings»Touch ID & Passcode* or *Settings»Face ID & Passcode.* The biometric authentication factor can be configured for device unlock, Apple Pay and iTunes and App Store.

The Passcode Policy Payload allows an Administrator to enable/disable modification of Touch ID through a Configuration Profile.

A passcode must be supplied for additional security validation in any of the following instances.

For Touch ID

- The device has just been turned on or restarted.
- For device software updates
- To wipe the device
- To view or change passcode settings
- To install iOS configuration profiles

For Face ID

- The device has just been turned on or restarted.
- The device hasn't been unlocked for more than 48 hours.
- The passcode hasn't been used to unlock the device in the last six and a half days and Face ID hasn't unlocked the device in the last 4 hours.
- The device has received a remote lock command.
- After five unsuccessful attempts to match a face
- After initiating power off/Emergency SOS by pressing and holding either volume button and the side button simultaneously for 2 seconds

3.4.7 X.509 Certificate Configuration

X.509 certificates are configured by an administrator using a Configuration Profile. See the [IOS_CFG] section *Certificate Payload* and section 3.1, *Configuration Profiles*, of this document for more details on Configuration Profiles.

Certificates have a certificate type that defines their respective application area. This ensures that only certificates defined for a specific application area are used. In addition, the database containing trust anchors for all certificates is protected via integrity check and write protection. The certificate types supported by the TOE are as follows.

- AppleX509Basic
- AppleSSL
- AppleSMIME
- AppleEAP
- AppleIPsec
- AppleCodeSigning
- AppleIDValidation
- AppleTimeStamping

Administrators can view all certificates on a device and remove any certificates it has installed via the MDM.

Users can manually remove certificates that have been installed on their device. Choose *Settings»General»Profile & Device Management*, select a profile, choose More Details, and choose the appropriate certificate to remove, unless the administrator has disallowed the removal of the Configuration Profile that contains the certificate.

When configuring the TOE to utilize EAP-TLS as part of a WPA2 protected Wi-Fi-network, the CA certificate(s) to which the server's certificate must chain can be configured using the PayLoadCertificateAnchorUUID key in the Wi-Fi payload of the Configuration Profile.

3.4.7.1 Certificate Validation

To configure the TOE to reject untrusted certificates, the administrator can use the TLSAllowTrustExceptions key in the Wi-Fi payload of the Configuration Profile which enforces that untrusted certificates are not accepted and the authentication fails if such untrusted certificates are presented.

To enforce the verification of the server name defined with the X.509 certificate during the WPA-EAP handshake between the TOE and the remote access point, the policy must contain the server name to be expected in the certificate with the TLSTrustedServerNames setting. This can be configured with the Apple Configurator when configuring the “Trust” of the certificates for Wi-Fi EAP configurations by adding the server name to the list of trusted servers.

Guidance and the API documentation related to certificate validation is provided in the “Certificate, Key, and Trust Services” [CKTSREF] in the section “Trust”. See the function ‘SecTrustEvaluate’.

3.4.8 Configure Enrollment of Mobile Device Into Management Configuration

The MDM server identity is provided to the Mobile Device by sending an MDM payload in a Configuration Profile.

The methods by which an MDM Agent can be enrolled are as follows.

- Device Enrollment Program (DEP)—Provides an automated and enforced method of automatically enrolling new devices
- Apple’s Profile Manager—Provides a manual method of enrolling devices
- Apple Configurator 2—Provides both automated and manual methods of enrolling devices
- Email or Website—Provides a way to distribute an enrollment profile

For the DEP, each MDM server must be registered with Apple at the MDM server DEP management portal. The DEP provides details about the server entity to identify it uniquely throughout the organization deploying the MDM server. Each server can be identified by either its system-generated universally unique identifier (UUID) or by a user-provided name assigned by one of the organization’s users. Both the UUID and server name must be unique within the organization.

For the DEP, an organization assigns iOS devices to Apple’s virtual MDM server using either Apple order numbers or device serial numbers. When the iOS device is powered on, the device will automatically connect to the virtual MDM server during setup and will be assigned to an MDM server specified in the MDM payload sent by the virtual MDM server to the iOS device. During the device enrollment, the MDM enrollment service returns a JavaScript Object Notation (JSON) dictionary to the mobile device with the keys shown in Table 8: Enrollment Keys.

Key	Value
server_name	An identifiable name for the MDM server
server_uuid	A system-generated server identifier
admin_id	Apple ID of the person who generated the current tokens that are in use

Key	Value
facilitator_id	Legacy equivalent to the admin_id key. This key is deprecated and may not be returned in future responses.
org_name	The organization name
org_email	The organization email address
org_phone	The organization phone
org_address	The organization address

Table 8: Enrollment Keys

Additional information on the DEP is provided in the [DEP_Guide].

For enrolling a device using Apple's Profile Manager, see the *Mobile Device Management* section of [PM_Help].

For enrolling a device using the Apple Configurator 2, see the *Prepare devices using automated enrollment* and *Prepare devices manually* sections of [AConfig].

For enrolling a device using other methods, this is specific to the MDM server being used. In general, the configuration profile is “made available” (e.g., through a website, email) to the iOS device, possibly via a link. The user clicks the link and starts the enrollment process.

3.5 Security Management Configuration

In the evaluated configuration, the TOE performs the management functions listed in Table *9*: Management Functions. These management functions can be managed either by the user or by an authorized administrator (marked by ‘X’) through an MDM system. Many of the administrator-performed management functions require the use of Configuration Profiles and/or MDM and/or Apple Configurator which are explicitly noted throughout this document. In addition, the Provided Guidance column in Table *9*: Management Functions references the section(s) in this document where guidance can be found to perform the respective management function. The management function values in parenthesis (e.g., F1, F2) in the following table correspond to the function values specified in Table 3 of the [ST].

Management Function	User	Administrator	When Enrolled	Provided Guidance
Configure password policy (F1)		X	X	Sections 3.1 and 3.4.1
Configure session locking policy (F2)		X	X	Sections 3.1 and 3.5.3
Enable/disable the VPN protection (F3)		X	X	Sections 3.1 and 3.3.2
Enable/disable Bluetooth, Wi-Fi, cellular radio (F4)	X			Section 3.4.4
Enable/disable cameras Enable/disable microphones (F5)	X X	X	X	Section 3.5.6
Transition to the locked state (F6)		X		Section 3.5.3
TSF wipe of protected data (F7)		X		Section 3.2.5

Management Function	User	Administrator	When Enrolled	Provided Guidance
Configure application installation policy by denying installation of applications (F8)		X	X	Section 3.1
Import keys/secrets into the secure key storage (F9)		X		Section 3.2.6
Destroy imported keys/secrets and no other keys/secrets in the secure key storage (F10)		X		Section 3.2.6
Import X.509v3 certificates in the Trust Anchor Database (F11)		X	X	Section 3.4.7
Remove imported X509v3 certificates and no other X509v3 certificates in the Trust Anchor Database (F12)	X	X	X	Section 3.4.7
Enroll the TOE in management (F13)	X			Section 3.5.10
Remove applications (F14)		X	X	Section 3.8
Update system software (F15)		X		Section 2.2
Install applications (F16)		X	X	Section 3.8
Remove Enterprise applications (F17)		X		Section 3.8
Configure the Bluetooth trusted channel[1] (F18)	X			Section 3.4.4
Enable/disable display notifications in the locked state of all notifications (F19)	X	X		Section 3.5.2
Enable/disable location services (F22)	X			Section 3.5.7
Enable/disable the use of Biometric Authentication Factor (F23)		X	X	Section 3.4.2
Wipe Enterprise data (F28)		X	X	Section 3.2.5

[1] There is no configuration for the Bluetooth trusted channel. It is secure by default.

Management Function	User	Administrator	When Enrolled	Provided Guidance
Configure whether to establish a trusted channel or disallow establishment if the TSF cannot establish a connection to determine the validity of a certificate (F30)		X	X	Section 3.4.7.1
Configure certificate used to validate digital signature on application (F33)		X	X	Sections 2.2 and 3.2.6
Configure the unlock banner (F36)		X	X	Section 3.5.5
Configure the auditable items (F37)		X	X	Section 3.6.3

Table 9: Management Functions

3.5.1 Install/Remove Apps from the TOE

The Administrator can install apps to the TOE using an MDM system or Apple Configurator. Refer to the [iOSDeployRef] section *app and book distribution* and the [AConfig] section *Add apps*.

If the device is enrolled in MDM Profile, managed apps on the device can be removed by an administrator remotely by the MDM, or when the user removes their own device from MDM Profile. Removing the app also removes the data associated with the removed app. For more information on managed apps refer to the “iOS Deployment Reference” [iOSDeployRef].

Users can install or remove an app from their TOE device. See the [iPhone_UG] and the [iPad_UG] *App Store* Section *Purchase, redeem, and download*.

3.5.2 Configure Access and Notification in Locked State

Access to certain optional features can be allowed when the TOE device is in locked state. These optional features include the following.

- Email notification
- Calendar appointment
- Text message notification

To allow access to these optional features when the TOE device is locked, go to *Settings»Touch ID &* Passcode (TOE devices with Touch ID) or *Settings»FaceID Passcode* (iPhone X) and select the features you want to allow access under the ‘Allow Access When Locked’ menu. Those items may be restricted by a Configuration Profile installed by an administrator. See section 3.1, *Configuration Profiles*, of this document for more information about Configuration Profiles.

Certain display notifications can be set when the TOE device is in the locked state. To enable/disable display notifications in the locked state, refer to the [iPad_UG] and the [iPhone_UG] *Basics* section *Notification*.

Alternatively, displaying notifications when in the locked state can be allowed or disallowed via the allowLockScreenNotificationsView key in the Restrictions Payload of a Configuration Profile. See section 3.1, *Configuration Profiles*, of this document for more information about Configuration Profiles.

3.5.3 Device/Session Locking

The TOE device is locked after a configurable time of user inactivity or upon request of the user. This time can be defined by an administrator using a Configuration Profile. That is, by setting the configuration key “maxInactivity” in the Passcode Policy Payload described in the [IOS_CFG] to the desirable time. See the sample Passcode Policy Payload in section 3.4.1, *Passcode Authentication Configuration*.

The user can set the time of user inactivity on their device by going to *Settings»General»Auto-Lock* and selecting the desired time interval.

The lock screen of a device can be defined and set for supervised devices by an administrator using Apple Configurator or an MDM.

3.5.4 Timestamp Configuration

To set the time on the TOE device, see the [iPad_UG] and the [iPhone_UG] *Get Started* section, *Date and time*. In the evaluated configuration, the TOE must be configured to update its time automatically.

3.5.5 TOE Banner Configuration

The TOE banner can be configured by creating a background picture with the relevant information and configuring that picture as the background for the lock screen. This can be performed using the Apple Configurator.

Alternatively, a notice and consent warning message can be configured through an app that provides the requisite notice and acknowledgement functionality rather than through iOS itself. The implementing organization must deploy a customizable app that provides users notice of the banner (e.g., through the Apple Push Notification Service) and also the ability to acknowledge the banner content within the app.

3.5.6 Enable/Disable Cameras and Microphones

The [iPad_UG] and the [iPhone_UG] *Privacy and security* section provides information for user to enable/disable cameras across the device and to enable/disable cameras and microphones on a per-app basis.

3.5.7 Enable/Disable Cellular, Wi-Fi, Wi-Fi Hotspot, Bluetooth

Users can enable/disable cellular by following instructions provided in the [iPhone_UG] and the [iPad_UG] *Safety, handling, and support* section *Cellular data settings*.

Users can enable/disable Wi-Fi by following the instructions provided in the [iPhone_UG] and the [iPad_UG] *Get started* section *Connect to Wi-Fi*.

User can enable/disable Wi-Fi hotspot by following the instructions provided in the [iPhone_UG] and the [iPad_UG] *Basics* section *iPad and other devices*, *Personal Hotspot*.

Users can enable/disable Bluetooth by following the instructions provided in the [iPhone_UG] and the [iPad_UG] *Basics* section *iPad and other devices*, *Connect bluetooth devices*.

3.5.8 Enable/Disable Location Services

Users can enable/disable location services by following the instructions provided in the [iPhone_UG] and the [iPad_UG] *Privacy and security* section.

3.5.9 Enable/Disable Remote Backup

Administrators can allow/disable remote backup for the TOE (e.g., to iCloud, iTunes) using a Configuration Profile. See section 3.1, *Introduction*, for more detail on Configuration Profiles.

Users can use enable/disable remote backup to iCloud or iTunes by following the instructions provided in the [iPhone_UG] *Get started, Back up iPhone* and the [iPhone_UG] *Get started* section *Back up iPhone*.

3.5.10 TOE Enrollment

Users and administrators can enroll the TOE device in management. Information for enrolling the TOE is provided in section *Configuration and management*, subsection *Mobile device management (MDM) of* the [iOSDeployRef].

A sample Configuration Profile for the aforementioned management functions is provided below.

```
<plist version="1.0">
<dict>
        <key>PayloadContent</key>
<array>
                <dict>
                        <key>PayloadDescription</key>
                        <string>Configures restrictions</string>
                        <key>PayloadDisplayName</key>
                        <string>Restrictions</string>
                        <key>PayloadIdentifier</key>
                        <string>… </string>
                        <key>PayloadType</key>
                        <string>com.apple.applicationaccess</string>
                        <key>PayloadUUID</key>
                        <string>…</string>
                        <key>PayloadVersion</key>
                        <integer>1</integer>
                        <key>allowAppRemoval</key>
                        <true/>
                        <key>allowAssistant</key>
                        <true/>
                        <key>allowAssistantWhileLocked</key>
                        <true/>
                        <key>allowCamera</key>
                        <false/>
<key>allowUntrustedTLSPrompt</key>
        <true/>
<key>forceEncryptedBackup</key>
        <true/>
<key>forceITunesStorePasswordEntry</key>
        <false/>
..
</dict>
</array>
<key>PayloadDisplayName</key>
<string>Profile for FMT_SMF_EXT.1</string>
<key>PayloadIdentifier</key>
        <string>… </string>
        <key>PayloadOrganization</key>
        <string>…</string>
        <key>PayloadRemovalDisallowed</key>
        <true/>
        <key>PayloadType</key>
        <string>Configuration</string>
        <key>PayloadUUID</key>
```

```
            <string>…</string>
            <key>PayloadVersion</key>
            <integer>1</integer>
</dict>
</plist>
```

Figure 1: Sample Configuration Profile

3.5.11 TOE Unenrollment Prevention

During the enrollment process, a Configuration Profile called an MDM Payload is loaded onto the device and used to associate the device to an MDM Server. If the MDM Payload is removed, the device will no longer be enrolled with the MDM Server.

As described in the “Configuration Profile Key Reference” [iOS_CFG], the MDM Server administrator can use the PayloadRemovalDisallowed key to allow or disallow the ability of a user to remove the MDM Payload from the device. It is up to the MDM Server to ensure that this key is set appropriately. The device must be in Supervised Mode to lock the MDM Payload to the device.

An MDM Payload can have a removal password associated with it. If the PayloadRemovalDisallowed key is set to prevent unenrollment and the MDM Payload has a removal password associated with it, the user can unenroll the device if the user knows the removal password.

3.6 Audit

3.6.1 Audit Logging

iOS logging capabilities collect a wide array of information concerning TOE usage and configuration. The available commands and responses constitute audit records and must be configured by TOE administrators using configuration profiles. The details for profile implementation and audit record collection can be located in [IOS_CFG] and [IOS_LOGS] respectively.

Each audit record, at a minimum, contains the following:

- date and time of the event;
- type of event (this is described as log level and log tag)
- subject identity (this is described as PID and PPID)
- the outcome (success or failure) of the event; and
- any applicable required additional information.

Each field of the example log below corresponds with the above format.

Date and Time	Type of event	Subject identity	The outcome
Dec 10 15:22:29.546196	<Error>:	iPadAir2 neagent[446]	Certificate authentication data could not be verified Failed to process IKE Auth packet

Figure 2: Example Audit Log

Table *10*: Audit Record Format provides examples of audit events required by MDFPP as well as the MDM Agent and WLAN extended packages.

ST Requirement	Auditable Events	Additional Audit Record Contents	Example of Audit Records
Device audit records			
FAU_GEN.1(1) (MDF)	Start-up and shutdown of the audit functions	No additional information.	Dec 5 11:34:58 iPhone-6s mdmd(libdyld.dylib)[6307] <Notice>: mdmd starting… Dec 5 11:39:19 iPhone-6s mdmd(libdyld.dylib)[6314] <Notice>: mdmd preparing to stop.
	All auditable events for the [*not selected]* level of audit	No additional information.	Dec 5 11:34:58 iPhone-6s mdmd(libdyld.dylib)[6307] <Notice>: mdmd starting…
	All administrative actions	No additional information.	Dec 5 12:30:48 iPhone-6s dmd[3038] <Notice>: Received request: <DMFInstallProfileRequest: 0x100c207f0>, from client: <CATTaskSession: 0x100c2f620 { state = Connected, session = BCD262D5-C3B1-4E1F-879C-900ADAFC490E, transport = <CATXPCTransport: 0x100c375b0 { state = Connected }> }>
	Start-up and shutdown of the Rich OS	No additional information.	Wed Jan 31 06:24:39 2018 iPhone com.apple.xpc.launchd[1] (com.apple.xpc.launchd.domain.system) <Notice>: System shutdown initiated by: SpringBoard.63 apcie[2:baseband-pcie]::waitForL2Entry timeout waiting for L2 Wed Jan 31 06:24:46 2018 iPhone com.apple.xpc.launchd[1] (com.apple.xpc.launchd.domain.system) <Notice>: Userspace teardown took: 6720 ms Wed Jan 31 06:24:46 2018 iPhone com.apple.xpc.launchd[1] (com.apple.xpc.launchd.domain.system) <Notice>: Will be calling reboot(2) with flags: 0x8 Kext loading now disabled. Kext unloading now disabled. Kext autounloading now disabled. syncing disks... Killing all processes flushed all txn's! apfs: total mem allocated: 29123031 (27 mb); all done. going home. (numMountedAPFSVolumes 2) kern_close_file_for_direct_io vnode_close(0) done CPU halted
FCS_STG_EXT.1 (MDF)	Import or key destruction	Identity of key. Role and identity of requestor.	Dec 5 13:38:59 iPhone-6s profiled[93] <Notice>: Removing profile \M-b\M^@\M^\com.apple.mdm.osxserver.atsec.local.9b7bc010-bc15-0135-1603-1801a79c5047.pushed\M-b\M^@\M^]... Dec 5 13:39:00 iPhone-6s profiled[93] <Notice>: Committing restrictions.

ST Requirement	Auditable Events	Additional Audit Record Contents	Example of Audit Records
FCS_STG_EXT.3 (MDF)	Failure to verify integrity of stored key.	Identity of key being verified.	Sep 25 09:19:38 iPhone securityd[96] <Notice>: Authentication is needed for genp,rowid=49 (-25330): Error Domain=NSOSStatusErrorDomain Code=-25330 "(null)" Sep 25 09:19:38 iPhone securityd[96] <Notice>: Authentication is needed a[280]/1#4 LF=0 copy_matching Error Domain=NSOSStatusErrorDomain Code=-25330 "(null)" Sep 25 09:19:38 iPhone securityd[96] <Notice>: ks_encrypt_data (db): failed: Sep 25 09:19:38 iPhone securityd[96] <Notice>: insert failed for item <genp,rowid=null,cdat=2017-09-25 07:19:38 +0000,mdat=2017-09-25 07:19:38 +0000,desc=null,icmt=null,crtr=null,type=null,scrp=null,labl=null,alis=null,invi=null,nega=null,cusi=null,prot=null,acct=,svce=key_3T,gena=null,agrp=B75W8GX8D3.test.a,pdmn=ck,sync=0,tomb=0,sha1=8756FB888E67AFBFA1E64470E1CED3445A636189,vwht=null,tkid=null,v_Data=<?>,v_pk=C7575E0532019C93D96E98F48036B2E68D89F0A2,accc=3120300A0C0470726F740C02636B30120C0361636C310B30090C046461636C010101,u_Tomb=null,musr=,UUID=4EF98D12-E6AF-4E50-B71D-533E99C7066F,sysb=null,pcss=null,pcsk=null,pcsi=null,persistref=> with Error Domain=NSOSStatusErrorDomain Code=-25330 "(null)" UserInfo={-25330=(… Sep 25 09:19:38 iPhone securityd[96] <Notice>: Authentication is needed a[280]/1#4 LF=0 add Error Domain=NSOSStatusErrorDomain Code=-25330 "(null)" UserInfo={-25330=(
FCS_TLSC_EXT.1 (WLAN)	Failure to establish an EAP-TLS session.	Reason for failure. Non-TOE endpoint connection.	12/11/17 15:34:37.442 <NOTICE>: Attempting Apple80211AssociateAsync 12/11/17 15:34:37.442 <NOTICE>: Attempting to join EAP network: test 12/11/17 15:34:37.442 <NOTICE>: Completed Apple80211AssociateAsync (-3900 - 0xFFFFF0C4) 12/11/17 15:34:37.442 <ERROR>: Failed to join(-3900 - 0xFFFFF0C4): test
	Establishment/ termination of an EAP-TLS session.	No additional information.	12/11/17 15:34:37.442 <NOTICE>: Attempting Apple80211AssociateAsync 12/11/17 15:34:37.442 <NOTICE>: Attempting to join EAP network: test 12/11/17 15:34:37.442 <NOTICE>: Completed Apple80211AssociateAsync (-3900 - 0xFFFFF0C4) 12/11/17 15:34:37.442 <ERROR>: Failed to join(-3900 - 0xFFFFF0C4): test
FDP_DAR_EXT.1 (MDF) FDP_DAR_EXT.2 (MDF)	Failure to encrypt/decrypt data. Failure to encrypt / decrypt data.	No additional information.	Sep 25 09:19:38 iPhone securityd[96] <Notice>: Authentication is needed for genp,rowid=49 (-25330): Error Domain=NSOSStatusErrorDomain Code=-25330 "(null)" Sep 25 09:19:38 iPhone securityd[96] <Notice>: Authentication is needed a[280]/1#4 LF=0 copy_matching Error Domain=NSOSStatusErrorDomain Code=-25330 "(null)" Sep 25 09:19:38 iPhone securityd[96] <Notice>: ks_encrypt_data (db): failed: Sep 25 09:19:38 iPhone securityd[96] <Notice>: insert failed for item <genp,rowid=null,cdat=2017-09-25 07:19:38 +0000,mdat=2017-09-25 07:19:38 +0000,desc=null,icmt=null,crtr=null,type=null,scrp=null,labl=null,alis=null,invi=null,nega=null,cusi=null,prot=null,acct=,svce=key_3T,gena=null,agrp=B75W8GX8D3.test.a,pdmn=ck,sync=0,tomb=0,sha1=8756FB888E67AFBFA1E64470E1CED3445A636189,vwht=null,tkid=null,v_Data=<?>,v_pk=C7575E0532019C93D96E98F48036B2E68D89F0A2,accc=3120300A0C0470726F740C02636B30120C0361636C310B30090C046461636C010101,u_Tomb=null,musr=,UUID=4EF98D12-E6AF-4E50-B71D-

ST Requirement	Auditable Events	Additional Audit Record Contents	Example of Audit Records
			533E99C7066F,sysb=null,pcss=null,pcsk=null,pcsi=null,persistref=> with Error Domain=NSOSStatusErrorDomain Code=-25330 "(null)" UserInfo={-25330=(… Sep 25 09:19:38 iPhone securityd[96] <Notice>: Authentication is needed a[280]/1#4 LF=0 add Error Domain=NSOSStatusErrorDomain Code=-25330 "(null)" UserInfo={-25330=(
FDP_STG_EXT.1 (MDF)	Addition or removal of certificate from Trust Anchor Database	Subject name of certificate	Dec 5 13:38:59 iPhone-6s profiled[93] <Notice>: Removing profile \M-b\M^@\M^\com.apple.mdm.osxserver.atsec.local.9b7bc010-bc15-0135-1603-1801a79c5047.pushed\M-b\M^@\M^]... Dec 5 13:39:00 iPhone-6s profiled[93] <Notice>: Committing restrictions.
FIA_X509_EXT.1 (MDF)	Failure to validate x.509v3 certificate	Reason for failure of validation	12/11/17 15:34:37.442 <ERROR>: Failed to join(-3900 - 0xFFFFF0C4): test
FPT_TST_EXT.1 (MDF)	Initiation of self-test	No additional information.	SEP: SEP: FIPS POST begin SEP: FIPSPOST_L4 fipspost_post:109: PASSED: (2 ms) - fipspost_post_integrity SEP: sks: FIPS POST Succeeded
FPT_TST_EXT.1	Failure of self-test	No additional information.	fipspost_post fipspost_post_integrity -POST_FAILURE: 0xFFFFFFFF
FPT_TST_EXT.2 (MDF)	Start-up of TOE	No additional information.	booting kernel at 0x87d703844
FPT_TST_EXT.1 (WLAN)	Execution of this set of TSF self-test.	No additional information.	corecrypto_kext_start called: tracing enabled FIPSPOST_KEXT fipspost_post:109: PASSED: (0 ms) - fipspost_post_integrity FIPSPOST_KEXT fipspost_post:115: PASSED: (0 ms) - fipspost_post_hmac FIPSPOST_KEXT fipspost_post:117: PASSED: (0 ms) - fipspost_post_aes_ecb FIPSPOST_KEXT fipspost_post:118: PASSED: (0 ms) - fipspost_post_aes_cbc FIPSPOST_KEXT fipspost_post:119: PASSED: (0 ms) - fipspost_post_aes_gcm FIPSPOST_KEXT fipspost_post:120: PASSED: (0 ms) - fipspost_post_aes_xts FIPSPOST_KEXT fipspost_post:121: PASSED: (0 ms) - fipspost_post_tdes_cbc FIPSPOST_KEXT fipspost_post:125: PASSED: (39 ms) - fipspost_post_rsa_sig FIPSPOST_KEXT fipspost_post:126: PASSED: (9 ms) - fipspost_post_ecdsa FIPSPOST_KEXT fipspost_post:127: PASSED: (2 ms) - fipspost_post_ecdh FIPSPOST_KEXT fipspost_post:128: PASSED: (0 ms) - fipspost_post_drbg_ctr FIPSPOST_KEXT fipspost_post:129: PASSED: (0 ms) - fipspost_post_drbg_hmac FIPSPOST_KEXT fipspost_post:136: all tests PASSED (129 ms)
FTA_WSE_EXT.1 (WLAN)	All attempts to connect to access points.	Identity of access point being connected to as well as success	12/11/17 15:34:37.442 <NOTICE>: Attempting Apple80211AssociateAsync 12/11/17 15:34:37.442 <NOTICE>: Attempting to join EAP network: test 12/11/17 15:34:37.442 <NOTICE>: Completed Apple80211AssociateAsync (-3900 - 0xFFFFF0C4) 12/11/17 15:34:37.442 <ERROR>: Failed to join(-3900 - 0xFFFFF0C4): test

ST Requirement	Auditable Events	Additional Audit Record Contents	Example of Audit Records
		and failures (including reason for failure)	
FTP_ITC_EXT.1 (WLAN)	All attempts to establish a trusted channel. Detection of modification of channel data.	Identification of the non-TOE endpoint of the channel.	12/11/17 14:00:53.955 <NOTICE>: Attempting Apple80211AssociateAsync 12/11/17 14:00:53.960 <NOTICE>: Attempting to join WPA network: testnet 12/11/17 14:00:54.232 <NOTICE>: Completed Apple80211AssociateAsync (0 - 0x0) 12/11/17 14:00:54.232 <NOTICE>: Joined: testnet 12/11/17 14:01:01.679 <NOTICE>: Update network <testnet>, requested by "configd"
Agent related audit records			
FAU_GEN.1(2) (AGENT)	Start-up and shutdown of the MDM Agent	No additional information.	Dec 5 11:34:58 iPhone-6s mdmd(libdyld.dylib)[6307] <Notice>: mdmd starting… Dec 5 11:39:19 iPhone-6s mdmd(libdyld.dylib)[6314] <Notice>: mdmd preparing to stop.
	Change in MDM policy	No additional information.	Dec 5 12:30:48 iPhone-6s profiled[93] <Notice>: Profile \M-b\M^@\M^\com.apple.mdm.osxserver.atsec.local.9b7bc010-bc15-0135-1603-1801a79c5047.pushed\M-b\M^@\M^] is replacing an existing profile having the same identifier.
	Any modification commanded by the MDM Server	No additional information.	Dec 5 12:30:46 iPhone-6s mdmd(ApplePushService)[6385] <Notice>: Received push notification.
FAU_ALT_EXT.2 (AGENT)	Type of alert.	No additional information.	Dec 5 11:34:58 iPhone-6s mdmd(libdyld.dylib)[6307] <Notice>: mdmd starting.
FAU_SEL.1 (AGENT)	All modifications to the audit configuration that occur while the audit collection functions are operating.	No additional information.	Apr 3 15:43:52 testers-iPhone **profiled[156]** <Notice>: Profile \M-b\M^@\M^\com.apple.config.testers-MacBook-Pro.local.mdm\M-b\M^@\M^] **removed.** Dec 5 13:38:59 iPhone-6s mdmd[6459] <Notice>: Attempting to perform Supervised request: RemoveProfile Dec 5 13:38:59 iPhone-6s mdmd(MDM)[6459] <Notice>: Handling request type: RemoveProfile
	Configuration that occur while the audit collection functions are operating.		Dec 5 13:38:59 iPhone-6s profiled[93] <Notice>: Removing profile \M-b\M^@\M^\com.apple.mdm.osxserver.atsec.local.9b7bc010-bc15-0135-1603-1801a79c5047.pushed\M-b\M^@\M^]... Dec 5 13:39:00 iPhone-6s profiled[93] <Notice>: Committing restrictions.

ST Requirement	Auditable Events	Additional Audit Record Contents	Example of Audit Records
FCS_TLSC_EXT.1 (AGENT)	Failure to establish a TLS session	Reason for failure.	12/11/17 15:34:37.442 <NOTICE>: Attempting Apple80211AssociateAsync 12/11/17 15:34:37.442 <NOTICE>: Attempting to join EAP network: test 12/11/17 15:34:37.442 <NOTICE>: Completed Apple80211AssociateAsync (-3900 - 0xFFFFF0C4) 12/11/17 15:34:37.442 <ERROR>: Failed to join(-3900 - 0xFFFFF0C4): test
	Failure to verify presented identifier	Presented identifier and reference identifier.	12/11/17 15:34:37.442 <NOTICE>: Attempting Apple80211AssociateAsync 12/11/17 15:34:37.442 <NOTICE>: Attempting to join EAP network: test 12/11/17 15:34:37.442 <NOTICE>: Completed Apple80211AssociateAsync (-3900 - 0xFFFFF0C4) 12/11/17 15:34:37.442 <ERROR>: Failed to join(-3900 - 0xFFFFF0C4): test
	Establishment/termination of a TLS session.	Non-TOE endpoint of connection.	Dec 5 14:22:42 iPhone-6 mdmd(CFNetwork)[6344] <Notice>: TIC TLS Event [1:0x101107b40]: 1, Pending(0)
FIA_ENR_EXT.2 (AGENT)	Enrollment in management	Reference identifier of MDM Server	Dec 5 12:30:48 iPhone-6s profiled[93] <Notice>: Checking for MDM installation... Dec 5 12:30:48 iPhone-6s profiled[93] <Notice>: ...finished checking for MDM installation.
FMT_POL_EXT.2 (AGENT)	Failure of policy validation.	Reason for failure of validation.	Dec 11 15:33:45 iPhone-5 wifid[41] <Notice>: WiFi:[534720825.769043]: Failed to join(-3906 - 0xFFFFF0BE): test Dec 11 15:33:45 iPhone-5 wifid[41] <Notice>: WiFi:[534720825.780136]: Failed to associate with test, reason -3906
FMT_SMF_EXT.3 (AGENT)	Success or failure of function.	No additional information.	Dec 5 12:30:46 iPhone-6s mdmd(ApplePushService)[6385] <Notice>: Received push notification.
FTP_ITC_EXT.1/ (AGENT)/ FTP_ITT_EXT.1 (AGENT)	Initiation and termination of trusted channel.	Trusted channel protocol. Non-TOE endpoint of connection.	12/11/17 14:00:53.955 <NOTICE>: Attempting Apple80211AssociateAsync 12/11/17 14:00:53.960 <NOTICE>: Attempting to join WPA network: testnet 12/11/17 14:00:54.232 <NOTICE>: Completed Apple80211AssociateAsync (0 - 0x0) 12/11/17 14:00:54.232 <NOTICE>: Joined: testnet

Table 10: Audit Record Format

3.6.2 Audit Storage

Audit records are not accessible to TOE Users, Administrators or MDM Administrators on the iOS device regardless of the device's configuration and cannot be modified in any way. All audit records can be synced to an MDM server using a configuration profile or manually via a trusted workstation using the Apple Configurator.

Depending on the underlying OS of the trusted workstation or MDM server, all of the device audit records are transferred to the following locations.

- macOS:
  - ~/Library/Logs/CrashReporter/MobileDevice/[Your_Device_Name]/
- Windows:
  - C:\Users\[Your_User_Name]\AppData\Roaming\AppleComputer\Logs\CrashReporter\MobileDevice\[Your_Device_Name]\

Audit records are not confined by a global capacity limit and are instead predefined individual services depending on what information is being captured. More information may be found in [IOS_LOGS].

There is no configuration required by the user. In other words, the users cannot configure where audit logs are stored as by default the audit logs are stored in the locations specified in this section.

3.6.3 Configure the Auditable Items

Audit record information is not available to TOE users or administrators on the TOE devices. The audit records are only accessible externally on trusted workstations via the Apple Configurator 2 or to an MDM server on enrolled devices.

[AConfig] describes how to use the device console to see all logged information that occurs between the device, Apple Configurator 2, and possibly connections outside your network to your mobile device management (MDM) solution or Apple. Administrators can mark a selection, clear the window to view a specific event, or save the log for troubleshooting.

According to [iOS_LOGS], additional logs can be specified by performing user actions on a device or through using a configuration profile. The table below shows which audit logs can be optionally gathered, and how they can be initiated.

Log type	Device user	Configuration Profile
3rd Party Apps for iOS	Instructions	
Accounts/AuthKit for iOS	Instructions	Profile
Ad Platforms for iOS	Instructions	Profile
AirTraffic for iOS	Instructions	
APNS (Apple Push Notification Service) for iOS	Instructions	Profile
App Store/iTunes Store for iOS	Instructions	Profile
Apple Pay for iOS	Instructions	Profile
Background Networking for iOS	Instructions	Profile
Baseband for iOS	Instructions	Profile
Battery Life for iOS	Instructions	Profile

Log type	Device user	Configuration Profile
Bluetooth for iOS	Instructions	Profile
Calendar/Reminders for iOS	Instructions	Profile
Carousel for iOS	Instructions	
CarPlay for iOS	Instructions	Profile
CFNetwork for iOS	Instructions	Profile
Charles Logs for iOS	Instructions	
CloudKit for iOS	Instructions	Profile
Console Logs for iOS	Instructions	
Contacts Data Export for iOS	Instructions	
Continuity (IDS) for iOS	Instructions	Profile
CoreMedia (HTTP Live Streaming) for iOS	Instructions	Profile
Crash Logs for iOS	Instructions	
Device-specific Information for iOS	Instructions	
Disk Space Diagnostics (FSMetadata) for iOS	Instructions	Profile
Enterprise SSO and Kerberos for iOS	Instructions	Profile
FaceTime for iOS	Instructions	Profile
Handoff for iOS	Instructions	
Health Database Extraction for iOS	Instructions	
HealthKit for iOS	Instructions	Profile
HomeKit for iOS	Instructions	Profile
iAP for iOS	Instructions	Profile
iCloud Backup for iOS	Instructions	Profile
iCloud Drive for iOS	Instructions	Profile
iCloud Key Value for iOS	Instructions	Profile
iCloud Photo Library for iOS	Instructions	Profile
iWork for iOS	Instructions	Profile
Location Services for iOS	Instructions	Profile
Mail for iOS	Instructions	Profile
Mail Raw Source for iOS	Instructions	
Mail Sync Diagnostics for iOS	Instructions	
Managed Configuration (MDM) for iOS	Instructions	Profile
Maps for iOS	Instructions	Profile
Media Player for iOS	Instructions	
Messages for iOS	Instructions	Profile

Log type	Device user	Configuration Profile
Multipeer Connectivity for iOS	Instructions	
Music for iOS	Instructions	
Phone (General) for iOS	Instructions	Profile
Photos Logging for iOS	Instructions	Profile
Podcasts for iOS	Instructions	
Screenshots and Screen Recordings for iOS	Instructions	
Siri for iOS	Instructions	Profile
Software Update for iOS	Instructions	Profile
Spotlight for iOS	Instructions	Profile
Stackshots for iOS	Instructions	
Sync Diagnostics (DataAccess) for iOS	Instructions	Profile
sysdiagnose for iOS	Instructions	Profile
Tailspin for iOS	Instructions	Profile
TCP Dump for iOS	Instructions	
Test Cases/Sample Projects for iOS	Instructions	
TestFlight for iOS	Instructions	Profile
Touch ID for iOS	Instructions	Profile
Unlock for iOS	Instructions	
Updater for iOS	Instructions	
VPN (Network Extension) for iOS	Instructions	Profile
Wallet for iOS	Instructions	Profile
Wi-Fi for iOS	Instructions	Profile

Table 11: Additional Audit Logs

3.7 Obtain Version Information

3.7.1 Obtain Operating System/Firmware Version

A device user can obtain information about the iOS software on the TOE device by following the instructions in the [iPad_UG] and the [iPhone_UG]. Go to *Settings»General»About*.

A device user can see all the installed apps on their device by going to the *Settings* menu on their device.

Administrators can query a variety of information about devices using an Mobile Device Management (MDM) server. This information includes hardware information, software information (such as iOS version), and a detailed list of all apps installed on the device. [DEP_GUIDE].

3.8 Installed Apps

Table *12*: Built-in Apps and Free Apps installed on TOE Devices lists the Built-in Apps and Free Apps installed on the TOE devices.

App Name	iPad	iPhone
App Store	X	X
Calculator		X
Calendar	X	X
Camera	X	X
Clips	X	
Clock	X	X
Compass		X
Contacts	X	X
FaceTime	X	X
Files	X	X
Find My Friends	X	X
Find My iPhone	X	X
Game Center		X
Garageband	X	
Extra	X	
Health		X
Home	X	X
iBooks	X	X
iMovie		
iTunes Store	X	X
iTunes U	X	X
Keynote	X	
Mail	X	X
Maps	X	X
Messages	X	X
Music	X	X
News	X	X
Notes	X	X
Numbers		X
Pages		X
Phone		X
Photo Booth	X	
Photos	X	X
Podcasts	X	X
Reminders	X	X
Safari	X	X
Settings	X	X
Stocks		X
Tips	X	X
TV	X	X
Videos		X
Voice Memos		X
Wallet		X
Watch		X
Weather		X

Table 12: Built-in Apps and Free Apps installed on TOE Devices

4 References

Table 1: TOE Guidance Documents, contains the references to the TOE guidance documents. Below are the references to the non-TOE guidance documents:

[BT] Specification of the Bluetooth System
https://www.bluetooth.com/specifications/adopted-specifications

[PP_MD_V3.1] U.S. Government Approved Protection Profile - Protection Profile for Mobile Device Fundamentals, Version 3.1
https://www.niap-ccevs.org/Profile/Info.cfm?id=417

[EP_MDM_AGENT_V3.0] U.S. Government Approved Protection Profile - Extended Package for Mobile Device Management Agents Version 3.0
https://www.niap-ccevs.org/Profile/Info.cfm?id=403

[PP_WLAN_CLI_EP_V1.0] Extended Package for WLAN Client Version 1.0
https://www.niap-ccevs.org/Profile/Info.cfm?id=386

[FACE_SEC] Face ID Security (November 2017)
https://images.apple.com/business/docs/FaceID_Security_Guide.pdf

[iOS_TEC] iOS Technology Overview
https://developer.apple.com/library/ios/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Introduction/Introduction.html

[CORECRYPTO_SECPOL]
– Apple CoreCrypto Cryptographic Module v8.0 for ARM. FIPS 140-2 Non-Proprietary Security Policy (February 2018)
https://csrc.nist.gov/projects/cryptographic-module-validation-program/Certificate/3148

and

– Apple CoreCrypto Kernel Cryptographic Module v8.0 for ARM. FIPS 140-2 Non-Proprietary Security Policy (February 2018)
https://csrc.nist.gov/projects/cryptographic-module-validation-program/Certificate/3147

5 Abbreviations and Acronyms

AES	Advanced Encryption Standard
API	Application Programming Interface
CA	Certificate of Authority
CBC	Cypher Block Chaining
CC	Common Criteria
CCM	Counter with CBC-MAC
CCMP	Counter Mode CBC-MAC Protocol
CDMA	Code Division Multiple Access
CDMA-EV-DO	Code Division Multiple Access Evolution-Data Optimized
CHAP	Challenge Handshake Authentication Protocol
DAR	Data-at-Rest
DC-HSDPA	Dual Carrier High-Speed Downlink Packet Access
DEK	Data Encryption Key
DEP	Device Enrollment Program
DES	Data Encryption Standard
EAP	Extensible Authentication Protocol
EAP-TLS	Extensible Authentication Protocol-Transport Layer Security
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
ECID	Electronic Chip ID
EDGE	Enhanced Data Rates for GSM Evolution
EP	Extended Package
FDD-LTE	Frequency-Division Duplex-Long Term Evolution
GCM	Galois/Counter Mode
GSM	Global System for Mobile Communications
HMAC	Keyed-Hash Message Authentication Code
HSPA+	High Speed Packet Access Plus
IKE	Internet Key Exchange
IV	Initialization Vector
JSON	JavaScript Object Notation
JTAG	Joint Test Action Group
KEK	Key Encryption Key
L2TP	Layer Two Tunneling Protocol
LLB	Low-Level Bootloader
LTE	Long-Term Evolution
MDF	Mobile Device Fundamentals
MDM	Mobile Device Management
MS-CHAP	Microsoft Challenge-Handshake Authentication Protocol
OTA	Over-the-Air
PAE	Port Access Entity
PBKDF	Password Based Key Derivation Function
PM	Profile Manager
PP	Protection Profile
PPTP	Point-to-Point Tunneling Protocol
REK	Root Encryption Key
RSA	Rivest-Shamir-Adleman
SCEP	Simple Certificate Enrollment Protocol
SEP	Secure Enclave Processor
SFR	Security Functional Requirement
SHA	Secure Hash Algorithm
SoC	System on a Chip
SSID	Service Set Identifiers
SSL	Secure Sockets Layer
ST	Security Target
TD-LTE	Time Division Long-Term Evolution
TD-SCDMA	Time Division Synchronous Code Division Multiple Access
TLS	Transport Layer Security
TN	Technical Note
TOE	Target of Evaluation
TSF	TOE Security Functionality
UMTS	Universal Mobile Telecommunications Service
UUID	Universally Unique Identifier
VPN	Virtual Private Network
WLAN	Wireless Local Area Network
WPA	Wi-Fi Protected Access
XML	Extensible Markup Language