



Assurance Activity Report

Version: 1.1
Date: 2018-03-30
Status: RELEASED
Classification: Public
Filename: VID10851_SER_AAR_Apple_iPad_and_iPhone_with_iOS_11.2_v1.1
Product: Apple iPad and iPhone Mobile Devices with iOS 11.2
Sponsor: Apple Inc.
Evaluation Facility: atsec information security corporation
Validation ID: 10851
Validation Body: NIAP CCEVS
Author(s): Trang Huynh, King Ables, Quentin Gouchet, Brandon Harvey
Quality Assurance: Fiona Pattinson

This report must not be used to claim product certification, approval, or endorsement by NIAP CCEVS, NVLAP, NIST, or any agency of the Federal Government.

atsec information security corporation
9130 Jollyville Road, Suite 260
Austin, TX 78759

Phone: +1 512-615-7300
Fax: +1 512-615-7301
www.atsec.com

Classification Note

Public Information (public)

This classification level is for information that may be made available to the general public. No specific security procedures are required to protect the confidentiality of this information. Information classified “public” may be freely distributed to anyone inside or outside of atsec.

Information with this classification shall be clearly marked “public”, except that it is not required to mark “public” on printed marketing material obviously intended for publication.

Revision History

Version	Date	Author(s)	Changes to Previous Revision	Application Notes
1.1	2018-03-30	King Ables	Address validator comments.	
1.0	2018-02-28	Trang Huynh	First version	

Table of Contents

1	Evaluation Basis and Documents	12
2	Evaluation Results	13
2.1	Security Functional Requirements	13
2.1.1	Security audit (FAU)	13
2.1.1.1	Audit Data Generation (FAU_GEN.1(1))	13
	TSS Assurance Activities	13
	Guidance Assurance Activities	13
	Test Activities	14
2.1.1.2	Audit Storage Protection (FAU_STG.1)	14
	TSS Assurance Activities	14
	Guidance Assurance Activities	15
	Test Activities	15
2.1.1.3	Prevention of Audit Data Loss (FAU_STG.4)	15
	TSS Assurance Activities	15
	Guidance Assurance Activities	15
	Test Activities	15
2.1.1.4	Audit Data Generation (FAU_GEN.1(2)(AGENT))	15
	FAU_GEN.1.1	15
	FAU_GEN.1.2	16
2.1.1.5	Extended: Agent Alerts (FAU_ALT_EXT.2(AGENT))	17
	FAU_ALT_EXT.2.1	17
	FAU_ALT_EXT.2.2	18
2.1.1.6	Security Audit Event Selection (FAU_SEL.1(2)(AGENT))	19
	TSS Assurance Activities	19
	Guidance Assurance Activities	19
	Test Activities	19
2.1.2	Cryptographic support (FCS)	20
2.1.2.1	Cryptographic Key Generation (FCS_CKM.1(1)/(2))	20
	TSS Assurance Activities	20
	Guidance Assurance Activities	20
	Test Activities	21
2.1.2.2	Extended: Cryptographic Key Support (REK) (FCS_CKM_EXT.1)	23
	TSS Assurance Activities	23
	Guidance Assurance Activities	24
	Test Activities	24
2.1.2.3	Cryptographic Key Establishment (FCS_CKM.2(1))	25
	TSS Assurance Activities	25
	Guidance Assurance Activities	25
	Test Activities	26
2.1.2.4	Cryptographic Key Establishment (While device is locked) (FCS_CKM.2(2))	28
	TSS Assurance Activities	28
	Guidance Assurance Activities	28
	Test Activities	28
2.1.2.5	Extended: Cryptographic Key Random Generation (FCS_CKM_EXT.2)	29

TSS Assurance Activities	29
Guidance Assurance Activities	29
Test Activities	29
2.1.2.6 Extended: Cryptographic Key Generation (FCS_CKM_EXT.3)	29
TSS Assurance Activities	29
Guidance Assurance Activities	30
Test Activities	30
2.1.2.7 Extended: Key Destruction (FCS_CKM_EXT.4)	33
TSS Assurance Activities	33
Guidance Assurance Activities	33
Test Activities	33
2.1.2.8 Extended: TSF Wipe (FCS_CKM_EXT.5)	34
TSS Assurance Activities	34
Guidance Assurance Activities	35
Test Activities	35
2.1.2.9 Extended: Salt Generation (FCS_CKM_EXT.6)	35
TSS Assurance Activities	35
Guidance Assurance Activities	36
Test Activities	36
2.1.2.10 Extended: Cryptographic Key Support (REK) (FCS_CKM_EXT.7)	36
TSS Assurance Activities	36
Guidance Assurance Activities	36
Test Activities	36
2.1.2.11 Cryptographic operation (Confidentiality Algorithms) (FCS_COP.1(1))	36
TSS Assurance Activities	36
Guidance Assurance Activities	36
Test Activities	36
2.1.2.12 Cryptographic operation (Hashing Algorithms) (FCS_COP.1(2))	40
TSS Assurance Activities	40
Guidance Assurance Activities	40
Test Activities	41
2.1.2.13 Cryptographic operation (Signature Algorithms) (FCS_COP.1(3))	42
TSS Assurance Activities	42
Guidance Assurance Activities	42
Test Activities	42
2.1.2.14 Cryptographic operation (Keyed Hash Algorithms) (FCS_COP.1(4))	43
TSS Assurance Activities	43
Guidance Assurance Activities	43
Test Activities	43
2.1.2.15 Cryptographic operation (Password-based Key Derivation Functions) (FCS_COP.1(5))	44
TSS Assurance Activities	44
Guidance Assurance Activities	44
Test Activities	44
2.1.2.16 Extended: HTTPS Protocol (FCS_HTTPS_EXT.1)	44
TSS Assurance Activities	44
Guidance Assurance Activities	45

Test Activities	45
2.1.2.17 Extended: Initialization Vector Generation (FCS_IV_EXT.1)	45
TSS Assurance Activities	45
Guidance Assurance Activities	45
Test Activities	45
2.1.2.18 Extended: Cryptographic Operation (Random Bit Generation) (FCS_RBG_EXT.1)	45
TSS Assurance Activities	45
Guidance Assurance Activities	46
Test Activities	46
2.1.2.19 Extended: Cryptographic Algorithm Services (FCS_SRV_EXT.1)	47
TSS Assurance Activities	47
Guidance Assurance Activities	47
Test Activities	47
2.1.2.20 Extended: Cryptographic Key Storage (FCS_STG_EXT.1)	47
TSS Assurance Activities	47
Guidance Assurance Activities	48
Test Activities	48
2.1.2.21 Extended: Encrypted Cryptographic Key Storage (FCS_STG_EXT.2)	49
FCS_STG_EXT.2.1	49
FCS_STG_EXT.2.2	49
2.1.2.22 Extended: Integrity of Encrypted Key Storage (FCS_STG_EXT.3)	50
TSS Assurance Activities	50
Guidance Assurance Activities	50
Test Activities	50
2.1.2.23 Extended: TLS Client Protocol (FCS_TLSC_EXT.1)	50
FCS_TLSC_EXT.1.1	50
FCS_TLSC_EXT.1.2	53
FCS_TLSC_EXT.1.3	54
FCS_TLSC_EXT.1.4	55
2.1.2.24 Extended: TLS Protocol (FCS_TLSC_EXT.2)	56
TSS Assurance Activities	56
Guidance Assurance Activities	56
Test Activities	56
2.1.2.25 Cryptographic Key Storage (FCS_STG_EXT.4(AGENT))	57
TSS Assurance Activities	57
Guidance Assurance Activities	57
Test Activities	57
2.1.2.26 Cryptographic Key Generation (FCS_CKM.1(2)(WLAN))	57
TSS Assurance Activities	57
Guidance Assurance Activities	58
Test Activities	58
2.1.2.27 Cryptographic Key Distribution (GTK) (FCS_CKM.2(3)(WLAN))	59
TSS Assurance Activities	59
Guidance Assurance Activities	59
Test Activities	59

2.1.2.28	Extensible Authentication Protocol-Transport Layer Security (FCS_TLSC_EXT.1(WLAN))	60
	TSS Assurance Activities	60
	Guidance Assurance Activities	61
	Test Activities	61
2.1.3	User data protection (FDP)	62
2.1.3.1	Extended: Security Access Control (FDP_ACF_EXT.1)	62
	FDP_ACF_EXT.1.1	62
	FDP_ACF_EXT.1.2	64
2.1.3.2	Extended: Protected Data Encryption (FDP_DAR_EXT.1)	65
	TSS Assurance Activities	65
	Guidance Assurance Activities	65
	Test Activities	66
2.1.3.3	Extended: Sensitive Data Encryption (FDP_DAR_EXT.2)	66
	FDP_DAR_EXT.2.1	66
	FDP_DAR_EXT.2.2	67
	FDP_DAR_EXT.2.3	68
	FDP_DAR_EXT.2.4	68
2.1.3.4	Extended: Subset Information Flow Control (FDP_IFC_EXT.1)	69
	TSS Assurance Activities	69
	Guidance Assurance Activities	69
	Test Activities	69
2.1.3.5	Extended: Storage of Critical Biometric Parameters (FDP_PBA_EXT.1)	70
	TSS Assurance Activities	70
	Guidance Assurance Activities	71
	Test Activities	71
2.1.3.6	Extended: User Data Storage (FDP_STG_EXT.1)	71
	TSS Assurance Activities	71
	Guidance Assurance Activities	71
	Test Activities	71
2.1.3.7	1 Extended: Inter-TSF User Data Transfer Protection (FDP_UPC_EXT.1)	72
	TSS Assurance Activities	72
	Guidance Assurance Activities	72
	Test Activities	73
2.1.4	Identification and authentication (FIA)	73
2.1.4.1	Extended: Authentication Failure Handling (FIA_AFL_EXT.1)	73
	TSS Assurance Activities	73
	Guidance Assurance Activities	74
	Test Activities	75
2.1.4.2	Extended: Bluetooth User Authorization (FIA_BLT_EXT.1)	75
	TSS Assurance Activities	75
	Guidance Assurance Activities	76
	Test Activities	77
2.1.4.3	Extended: Bluetooth Mutual Authentication (FIA_BLT_EXT.2)	77
	TSS Assurance Activities	77
	Guidance Assurance Activities	77

Test Activities	77
2.1.4.4 Rejection of Duplicate Bluetooth Connections (FIA_BLT_EXT.3)	78
TSS Assurance Activities	78
Guidance Assurance Activities	78
Test Activities	78
2.1.4.5 Extended: Secure Simple Pairing (FIA_BLT_EXT.4)	78
TSS Assurance Activities	78
Guidance Assurance Activities	79
Test Activities	79
2.1.4.6 Extended: Password Management (FIA_PMG_EXT.1)	79
TSS Assurance Activities	79
Guidance Assurance Activities	79
Test Activities	79
2.1.4.7 Extended: Authentication Throttling (FIA_TRT_EXT.1)	80
TSS Assurance Activities	80
Guidance Assurance Activities	80
Test Activities	80
2.1.4.8 Multiple Authentication Mechanisms (FIA_UAU.5)	80
TSS Assurance Activities	80
Guidance Assurance Activities	81
Test Activities	81
2.1.4.9 Extended: Accuracy of Biometric Authentication (FIA_BMG_EXT.1)	81
FIA_BMG_EXT.1.1	81
FIA_BMG_EXT.1.2	82
2.1.4.10 Extended: Biometric Enrollment (FIA_BMG_EXT.2)	83
TSS Assurance Activities	83
Guidance Assurance Activities	83
Test Activities	84
2.1.4.11 Extended: Biometric Verification (FIA_BMG_EXT.3)	84
TSS Assurance Activities	84
Guidance Assurance Activities	85
Test Activities	85
2.1.4.12 Extended: Handling Unusual Biometric Templates (FIA_BMG_EXT.5)	85
TSS Assurance Activities	85
Guidance Assurance Activities	86
Test Activities	86
2.1.4.13 Re-Authentication (FIA_UAU.6)	86
FIA_UAU.6.1	86
FIA_UAU.6.2	86
2.1.4.14 Protected Authentication Feedback (FIA_UAU.7)	87
TSS Assurance Activities	87
Guidance Assurance Activities	87
Test Activities	88
2.1.4.15 Extended: Authentication for Cryptographic Operation (FIA_UAU_EXT.1)	
.....	88
TSS Assurance Activities	88
Guidance Assurance Activities	89

Test Activities	89
2.1.4.16 Extended: Timing of Authentication (FIA_UAU_EXT.2)	90
TSS Assurance Activities	90
Guidance Assurance Activities	90
Test Activities	90
2.1.4.17 Extended: Validation of Certificates (FIA_X509_EXT.1)	90
TSS Assurance Activities	90
Guidance Assurance Activities	91
Test Activities	91
2.1.4.18 Extended: X509 Certificate Authentication (FIA_X509_EXT.2)	91
TSS Assurance Activities	91
Guidance Assurance Activities	92
Test Activities	92
2.1.4.19 Extended: Request Validation of certificates (FIA_X509_EXT.3)	93
TSS Assurance Activities	93
Guidance Assurance Activities	93
Test Activities	93
2.1.4.20 Extended: Enrollment of Mobile Device into Management (FIA_ENR_EXT.2(AGENT))	93
TSS Assurance Activities	93
Guidance Assurance Activities	94
Test Activities	94
2.1.4.21 Port Access Entity Authentication (FIA_PAE_EXT.1(WLAN))	95
TSS Assurance Activities	95
Guidance Assurance Activities	95
Test Activities	95
2.1.4.22 X.509 Certificate Authentication (EAP-TLS) (FIA_X509_EXT.2(WLAN))	95
TSS Assurance Activities	95
Guidance Assurance Activities	96
Test Activities	96
2.1.5 Security management (FMT)	97
2.1.5.1 Extended: Management of Security Functions Behavior (FMT_MOF_EXT.1)	97
FMT_MOF_EXT.1.1	97
FMT_MOF_EXT.1.2	97
2.1.5.2 Extended: Specification of Management Functions (FMT_SMF_EXT.1)	98
TSS Assurance Activities	98
Guidance Assurance Activities	104
Test Activities	108
2.1.5.3 Extended: Specification of Specification of Remediation Actions (FMT_SMF_EXT.2)	121
TSS Assurance Activities	121
Guidance Assurance Activities	121
Test Activities	121
2.1.5.4 Trusted Policy Update (FMT_POL_EXT.2(AGENT))	121
FMT_POL_EXT.2.1	121
FMT_POL_EXT.2.2	122

2.1.5.5	Specification of Management Functions (FMT_SMF_EXT.3(AGENT))	123
FMT_SMF_EXT.3.1		123
FMT_SMF_EXT.3.2		124
2.1.5.6	User Unenrollment Prevention (FMT_UNR_EXT.1(AGENT))	125
TSS Assurance Activities		125
Guidance Assurance Activities		126
Test Activities		126
2.1.5.7	Specification of Management Functions (Wireless LAN) (FMT_SMF_EXT.1(WLAN))	127
TSS Assurance Activities		127
Guidance Assurance Activities		127
Test Activities		127
2.1.6	Protection of the TSF (FPT)	127
2.1.6.1	Extended: Anti-Exploitation Services (ASLR) (FPT_AEX_EXT.1)	127
TSS Assurance Activities		127
Guidance Assurance Activities		128
Test Activities		128
FPT_AEX_EXT.1.4		128
2.1.6.2	Extended: Anti-Exploitation Services (Memory Page Permissions) (FPT_AEX_EXT.2)	129
FPT_AEX_EXT.2.1		129
FPT_AEX_EXT.2.2		129
2.1.6.3	Extended: Anti-Exploitation Services (Overflow Protection) (FPT_AEX_EXT.3)	130
TSS Assurance Activities		130
Guidance Assurance Activities		130
Test Activities		130
2.1.6.4	Extended: Domain Isolation (FPT_AEX_EXT.4)	130
TSS Assurance Activities		130
Guidance Assurance Activities		132
Test Activities		132
2.1.6.5	Extended: JTAG Disablement (FPT_JTA_EXT.1)	132
TSS Assurance Activities		132
Guidance Assurance Activities		133
Test Activities		133
2.1.6.6	Extended: Key Storage (FPT_KST_EXT.1)	133
TSS Assurance Activities		133
Guidance Assurance Activities		134
Test Activities		134
2.1.6.7	Extended: No Key Transmission (FPT_KST_EXT.2)	134
TSS Assurance Activities		134
Guidance Assurance Activities		135
Test Activities		135
2.1.6.8	Extended: No Plaintext Key Export (FPT_KST_EXT.3)	135
TSS Assurance Activities		135
Guidance Assurance Activities		136
Test Activities		136

2.1.6.9	Extended: Self-Test Notification (FPT_NOT_EXT.1)	136
	TSS Assurance Activities	136
	Guidance Assurance Activities	136
	Test Activities	136
	FPT_NOT_EXT.1.1	136
2.1.6.10	Reliable Time Stamps (FPT_STM.1)	137
	TSS Assurance Activities	137
	Guidance Assurance Activities	137
	Test Activities	138
2.1.6.11	Extended: TSF Cryptographic Functionality Testing (FPT_TST_EXT.1)	138
	TSS Assurance Activities	138
	Guidance Assurance Activities	138
	Test Activities	138
2.1.6.12	Extended: TSF Integrity Testing (FPT_TST_EXT.2)	139
	TSS Assurance Activities	139
	Guidance Assurance Activities	139
	Test Activities	139
2.1.6.13	Extended: Trusted Update: TSF Version Query (FPT_TUD_EXT.1)	140
	TSS Assurance Activities	140
	Guidance Assurance Activities	140
	Test Activities	140
2.1.6.14	Extended: Trusted Update Verification (FPT_TUD_EXT.2)	141
	TSS Assurance Activities	141
	Guidance Assurance Activities	141
	Test Activities	141
	FPT_TUD_EXT.2.3	141
	FPT_TUD_EXT.2.4	143
2.1.6.15	TSF Cryptographic Functionality Testing (Wireless LAN) (FPT_TST_EXT.1(WLAN))	143
	TSS Assurance Activities	143
	Guidance Assurance Activities	144
	Test Activities	146
2.1.7	TOE access (FTA)	146
2.1.7.1	Extended: TSF- and User-initiated Locked State (FTA_SSL_EXT.1)	146
	TSS Assurance Activities	146
	Guidance Assurance Activities	147
	Test Activities	147
2.1.7.2	Default TOE Access Banners (FTA_TAB.1)	147
	TSS Assurance Activities	147
	Guidance Assurance Activities	148
	Test Activities	148
2.1.7.3	Wireless Network Access (FTA_WSE_EXT.1(WLAN))	148
	TSS Assurance Activities	148
	Guidance Assurance Activities	148
	Test Activities	149
2.1.8	Trusted path/channels (FTP)	149
2.1.8.1	Trusted Channel Communication (FTP_ITC_EXT.1(2)(AGENT))	149

TSS Assurance Activities	149
Guidance Assurance Activities	149
Test Activities	150
2.1.8.2 Trusted Channel Communication (Wireless LAN) (FTP_ITC_EXT.1(WLAN))	
.....	150
TSS Assurance Activities	150
Guidance Assurance Activities	151
Test Activities	152
2.2 Security Assurance Requirements	153
2.2.1 Life-cycle support (ALC)	153
2.2.1.1 Labelling of the TOE (ALC_CMC.1)	153
2.2.1.2 TOE CM coverage (ALC_CMS.1)	154
2.2.1.3 Extension: Timely Security Updates (ALC_TSU_EXT.1)	155
2.2.2 Security Target evaluation (ASE)	157
2.2.3 Guidance documents (AGD)	157
2.2.3.1 Operational user guidance (AGD_OPE.1)	157
2.2.3.2 Preparative procedures (AGD_PRE.1)	160
2.2.4 Tests (ATE)	160
2.2.4.1 Independent testing - conformance (ATE_IND.1)	160
2.2.5 Vulnerability assessment (AVA)	161
2.2.5.1 Vulnerability survey (AVA_VAN.1)	161
A Appendixes	163
A.1 References	163
A.2 Glossary	167

List of Tables

Table 1: Notations used in SP 800-108 and SP 800-56C	31
Table 2: EAP-TLS Ciphersuites	61

1 Evaluation Basis and Documents

This evaluation is based on the "Common Criteria for Information Technology Security Evaluation" Version 3.1 Revision 5 [CC], the "Common Methodology for Information Technology Security Evaluation" [CEM] and the additional assurance activities given in the Protection Profile for Mobile Device Fundamentals (MDFPP) Version 3.1 [PP_MD_V3.1]; U.S. Government Approved Protection Profile - Extended Package for Mobile Device Management Agents Version 3.0 [EP_MDM_AGENT_V3.0], and General Purpose Operating Systems Protection Profile/ Mobile Device Fundamentals Protection Profile Extended Package (EP) Wireless Local Area Network (WLAN) Clients Version 1.0 [PP_WLAN_CLI_EP_V1.0] and associated technical decisions.

This evaluation claims Exact Compliance with the above US Government PP and EP's.

The following scheme documents and interpretations have been considered:

- [CCEVS-TD0194]: "Update to Audit of FTP_ITC_EXT.1/WLAN", version as of 2017-04-11.
- [CCEVS-TD0236]: "FCS_TLSC_EXT.2.1 - TLS Client Curve Limitation", version as of 2017-09-08.
- [CCEVS-TD0237]: "FAU_GEN.1.1(2) - FMT_UNR_EXT.1 Audit Record Selection-Based", version as of 2017-09-26.
- [CCEVS-TD0244]: "FCS_TLSC_EXT - TLS Client Curves Allowed", version as of 2017-11-16.

2 Evaluation Results

2.1 Security Functional Requirements

2.1.1 Security audit (FAU)

2.1.1.1 Audit Data Generation (FAU_GEN.1(1))

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-FAU_GEN.1-AGD-01

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

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

Summary

[CCGUIDE] [\[1\]](#) section 3.6 *Audit* describes auditing and provides Table 10 listing the audit events corresponding with the requirements specified in [ST] [\[1\]](#) with the associated audit events for those requirements from [PP_MD_V3.1] [\[1\]](#), [EP_MDM_AGENT_V3.0] [\[1\]](#), and [PP_WLAN_CLI_EP_V1.0] [\[1\]](#).

Section 3.6 also provides an example audit record and how the fields in this audit record map to the required fields specified in FAU_GEN.1. The example outlines the basic format for each audit record showing that the audit records include at minimum the following information:

- 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.

Table 10 from [CCGUIDE] [\[1\]](#) provides a list of audit events required by [PP_MD_V3.1] [\[1\]](#), [EP_MDM_AGENT_V3.0] [\[1\]](#), and [PP_WLAN_CLI_EP_V1.0] [\[1\]](#) including any additional audit record information as well as the outcome of the generated audit records.

As stated at the beginning of section 3.6, the available commands and responses constitute audit records and must be configured by the TOE administrators using configuration profiles. The details for profile implementation and audit record collection are documented in [IOS_CFG] [\[1\]](#) and [IOS_LOGS] [\[1\]](#). The evaluator examined these documents along with [ST] [\[1\]](#) and [CCGUIDE] [\[1\]](#) as well as through testing, the evaluator determined that the guidance contains all the administrative

actions and their associated audit events that are relevant to [PP_MD_V3.1] and [EP_MDM_AGENT_V3.0], and [PP_WLAN_CLI_EP_V1.0] and through use of the TOE. These administrative actions are found to be consistent with the actions specified in [ST].

The evaluator made the following observations:

- Every audit described in Table 10 provides the required information such as timestamps and subject identity.
- The SFRs covered in Table 10 match with Table 3 and Table 4 of [ST] that together cover the mandatory auditable events required by [PP_MD_V3.1], [EP_MDM_AGENT_V3.0], and [PP_WLAN_CLI_EP_V1.0]
- Each SFR from Table 3 and Table 4 of [ST] is covered by at least one audit record and vice versa.

Test Activities

Assurance Activity AA-FAU_GEN.1-ATE-01

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

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

Summary

The TOE devices do not provide an interface for the user to examine audit logs on the device. Audit records can only be accessed from an external MDM server or workstation running the Apple Configurator. Audit log information was provided by the developer. The evaluator examined audit output evidence and concluded it shows the required audit records are generated.

2.1.1.2 Audit Storage Protection (FAU_STG.1)

TSS Assurance Activities

Assurance Activity AA-FAU_STG.1-ASE-01

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

Summary

Section 8.9.1 in the [ST] describes the *Audit Records*.

The evaluator ensured that section 8.9.1 defines the storage location for the audit records. The TSS states that the storage location depends on the underlying platform used by the trusted workstation or MDM OS:

- macOS: ~/Library/Logs/CrashReporter/MobileDevice/[Your_Device_Name]/

- Windows:C:\Users\[Your_User_Name]\AppData\Roaming\Apple Computer\Logs\CrashReporter\MobileDevice\[Your_Device_Name]

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FAU_STG.1-ATE-01

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

Summary

During the test for FAU_GEN.1, the evaluator observed that the external interface to obtain the audit log information is read-only. This means that the interface is not capable of allowing a modify/write/delete operation. The audit trail is stored by the TOE such that the management application can only access the provided interface allowing the read operation.

2.1.1.3 Prevention of Audit Data Loss (FAU_STG.4)

TSS Assurance Activities

Assurance Activity AA-FAU_STG.4-ASE-01

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

Summary

Section 8.9.1 of [ST] [\[redacted\]](#) describes the *Audit Records*.

The audit storage capacity is defined by the profiles configured by the TOE administrators, as well as the default action when the storage capacity is reached.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.1.4 Audit Data Generation (FAU_GEN.1(2)(AGENT))

FAU_GEN.1.1


TSS Assurance Activities

Assurance Activity AA-FAU_GEN.1.1-2-MDMA-ASE-01

The evaluator shall check the TSS and ensure that it provides a format for audit records. Each audit record format type must be covered, along with a brief description of each field.

Summary

Section 8.9.1 *Audit Records* in the [ST]  describes how the TOE performs auditing.

Section 8.9.1 specifies how the audit record storage is performed by the TOE. The evaluator checked that section 8.9.1 describes how audit records are performed by the TOE. The TOE administrators must use configuration profiles to configure the audit storage capacity and the action to take when that limit is reached. The format of the audit record is described in the operational guidance, [CCGUIDE] .

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FAU_GEN.1.1-2-MDMA-ATE-01

The evaluator shall use the TOE to perform the auditable events defined in Table 1 and observe that accurate audit records are generated with contents and formatting consistent with those described in the TSS. Note that this testing can be accomplished in conjunction with the testing of the security mechanisms directly.

Summary

The tests are included in testing for FAU_GEN.1.

FAU_GEN.1.2


TSS Assurance Activities

Assurance Activity AA-FAU_GEN.1.2-2-MDMA-ASE-01

The evaluator shall check the TSS and ensure that it provides a format for audit records. Each audit record format type must be covered, along with a brief description of each field.

Summary

Section 8.9.1 *Audit Records* in the [ST]  describes how the TOE performs auditing.

Section 8.9.1 specifies how the audit record storage is performed by the TOE. The evaluator checked that section 8.9.1 describes how audit records are performed by the TOE. The TOE administrators must use configuration profiles to configure the audit storage capacity and the action to take when that limit is reached. The format of the audit record is described in the operational guidance, [CCGUIDE] .

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FAU_GEN.1.2-2-MDMA-ATE-01

When verifying the test results, the evaluator shall ensure the audit records generated during testing match the format specified in the administrative guide, and that the fields in each audit record have the proper entries.

Note that the testing here can be accomplished in conjunction with the testing of the security mechanisms directly.

Summary

The tests are included in testing for FAU_GEN.1.

2.1.1.5 Extended: Agent Alerts (FAU_ALT_EXT.2(AGENT))

FAU_ALT_EXT.2.1

TSS Assurance Activities

Assurance Activity AA-FAU_ALT_EXT.2.1-MDMA-ASE-01

The evaluator shall examine the TSS and verify that it describes how the alerts are implemented.

The evaluator ensures that the TSS describes how the candidate policy updates are obtained; and the actions that take place for successful (policy update installed) and unsuccessful (policy update not installed) cases. The software components that are performing the processing must also be identified in the TSS and verified by the evaluator.


The evaluator also ensures that the TSS describes how reachability events are implemented, and if configurable are selected in FMT_SMF_EXT.3.2. The evaluator verifies that this description clearly indicates who (MDM Agent or MDM Server) initiates reachability events.

Summary

Section 8.9.2 *MDM Agent Alerts* in the [ST]  describes the agent alerts generated and received by the TOE.

Section 8.9.2 describes that the MDM agent generates and sends an alert in response to an MDM server request. Section 8.9.2.2 describes how the agent communicates to the server that a policy was (un)successfully applied. Namely,

"When the application of policies to a mobile device is successful the MDM Agent replies with an MDM Result Payload with Status value "Acknowledged". If a policy update is not successfully installed then the MDM Agent replies with an MDM Result Payload with Status value "Error" or CommandFormatError, "Idle" and "NotNow"."

Section 8.9.2.3 describes alerts on receiving periodic reachability events and that periodic reachability events are initiated by the MDM Server using Push Notifications. The software component processing these alerts is the Apple MDM Agent described in [iOS_MDM] .

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FAU_ALT_EXT.2.1-MDMA-ATE-01

Test 1: The evaluator shall perform a policy update from the test environment MDM server. The evaluator shall verify the MDM Agent accepts the update, makes the configured changes, and reports the success of the policy update back to the MDM Server.

Test 2: The evaluator shall perform each of the actions listed in FAU_ALT_EXT. 1.1 and verify that the alert does in fact reach the MDM Server.

Test 3: The evaluator shall configure the MDM Agent to perform a network reachability test, both with and without such connectivity and ensure that results reflect each.

Summary

For test 1, the evaluator triggered a policy update by changing the minimum passcode length and then deploying the policy to the TOE using the Profile Manager and verified that the TOE was reconfigured.

For test 2, using the same test, the evaluator verified that the Profile Manager received notification of the update.

For test 3, the evaluator repeated the same test but disconnected the TOE from WLAN first. The evaluator confirmed the notification was automatically send upon reconnection of the TOE to the WLAN.

FAU_ALT_EXT.2.2

TSS Assurance Activities

Assurance Activity AA-FAU_ALT_EXT.2.2-MDMA-ASE-01

The evaluator shall ensure that the TSS describes under what circumstances, if any, the alert may not be generated (e.g., the device is powered off or disconnected from the trusted channel), how alerts are queued, and the maximum amount of storage for queued messages.

Summary

Section 8.9.2.1 in the [ST] describes the *Queuing of Alerts*.

Section 8.9.2.1 describes that if alerts cannot be sent by the device (for example, if the device is out of reach of a network), these alerts will be queued. This section explains that the queue cannot become long, since when the device is out of communication with the MDM, no additional MDM requests can be received. Whenever the device is not able to perform an MDM server request (for example, databases cannot be modified when the device is locked with Data Protection), the device will issue the NotNow status without performing the command. After issuing the NotNow status, the device will poll the server later on. The device will poll the server until a successful transaction is completed.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FAU_ALT_EXT.2.2-MDMA-ATE-01

The evaluator shall remove network connectivity from the MDM Agent and generate an alert/event as defined in FAU_ALT_EXT.2.1. The evaluator shall restore network connectivity to the MDM Agent and verify that the alert generated while the TOE was disconnected is sent by the MDM Agent upon re-establishment of the connectivity.

Summary

The evaluator disconnected the TOE from the WLAN and performed the steps in test 1 from FAU_ALT_EXT.2.1 using a different minimum password length. He verified the alert is held until the TOE was reconnected to the WLAN, then within a minute, it was sent and the new password length configured.

2.1.1.6 Security Audit Event Selection (FAU_SEL.1(2)(AGENT))

TSS Assurance Activities


No assurance activities defined.



Guidance Assurance Activities

Assurance Activity AA-FAU_SEL.1-2-MDMA-AGD-01

The evaluator shall examine the operational guidance to determine that it contains instructions on how to define the set of auditable events as well as explains the syntax for multi-value selection (if applicable). The evaluator shall also verify that the operational guidance shall identify those audit records that are always recorded, regardless of the selection criteria currently being enforced.

Summary

[CCGUIDE]  section 3.6.3 *Select the auditable items* provides guidance for how to select auditable events to be audited. It states the following:

- 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.
- Per [IOS_LOGS] , additional logs can be specified by performing user actions on a device or through using a configuration profile.
- Table 11 "Additional audit logs" identifies all the logs that can be optionally selected and how they can be initiated, the majority of which requires a configuration profile.

Test Activities

Assurance Activity AA-FAU_SEL.1-2-MDMA-ATE-01

Test 1: For each attribute listed in the requirement, the evaluator shall devise a test to show that selecting the attribute causes only audit events with that attribute (or those that are always recorded, as identified in the administrative guidance) to be recorded.

Test 2 [conditional]: If the TSF supports specification of more complex audit pre-selection criteria (e.g., multiple attributes, logical expressions using attributes) then the evaluator shall devise tests showing that this capability is correctly implemented. The evaluator shall also, in the test plan, provide a short narrative justifying the set of tests as representative and sufficient to exercise the capability.

Summary

The TOE devices do not provide an interface for the user to examine audit logs. Audit records are only accessible via an external management platform. Audit log information was provided by the developer. The evaluator examined audit output evidence and concluded it shows that individual audit attributes may be selected.

2.1.2 Cryptographic support (FCS)

2.1.2.1 Cryptographic Key Generation (FCS_CKM.1(1)/(2))

TSS Assurance Activities

Assurance Activity AA-FCS_CKM.1-ASE-01

The evaluator shall ensure that the TSS identifies the key sizes supported by the TOE. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme.

Summary

Section 8.10 of the [ST] [\[ST\]](#) specifies the *Mapping to the Security Functional Requirements*.

The table entry FCS_CKM.1(1)(MDF) in section 8.10 specifies the key sizes for FCS_CKM.1.1:

"The modules can generate RSA key pairs with modulus sizes of 2048 to 3072 in increments of 32 bits. ECC key pairs can be generated for NIST curves P-256, P-384."

Guidance Assurance Activities

Assurance Activity AA-FCS_CKM.1-AGD-01

The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key generation scheme(s) and key size(s) for all uses defined in this PP.

Summary

The developer provided the following document as the main administrator guide for configuring the TOE in the evaluated configuration:

- 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 [CCGUIDE] [\[CCGUIDE\]](#)

The Common Criteria Guide ([CCGUIDE] [\[CCGUIDE\]](#)) provides clarifications and changes to the Apple documentation and should 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.

The SFR components FCS_TLSC_EXT.1(MDF) and FCS_TLSC_EXT.1/WLAN (WLAN) in [ST] [\[ST\]](#) define the TLS cipher suites supported by the TOE. No configuration method to alter these cipher suites is provided. Also, the TOE generates ECC curve 25519 keys when protecting files that may need to be updated when the device is locked as described in the section "NSFileProtectionCompleteUnlessOpen" in the TSS of [ST] [\[ST\]](#). The 256-bit key size is defined by the curve and is not configurable.

The TOE generates RSA and NIST curve ECC keys when establishing a protected communications channel with TLS. The algorithm and key size used depends on the negotiation between the two endpoints. Cipher suites supported by the TOE are listed in [CCGUIDE] [\[CCGUIDE\]](#) Table 5 and Table 6 for EAP-TLS and TLS, respectively.

[CCGUIDE] [\[1\]](#) section 3.2 *Crypto-Related Function Configuration* states that the TOE comes with two cryptographic modules that provide cryptographic support for the TOE: Apple iOS CoreCrypto Kernel Module v8 for ARM and Apple iOS CoreCrypto Module v8 for ARM.

Also, [CCGUIDE] [\[1\]](#) section 3.2.1 *Key Generation, Signature Generation and Verification* states that 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 key size of 256 bits

[CCGUIDE] [\[1\]](#) Table 3 entry FCS_CKM.1(1) (MDF) states that no configuration is needed for key generation as the TOE provides programming interface (API) that allows specification of the requested key sizes and key types. Furthermore, section 3.2.1 also states that the TOE provides a programming interface (API) for key pair generation (function SecKeyGeneratePair), described in the 'Certificate, Key, and Trust Services Reference' [CKTSREF] [\[1\]](#) in section Keys. The 'kSecAttrKeyType' key passed in with the 'parameters' parameter defines the type of key pair and the 'kSecAttrKeySizeInBits' key defines the key size. Defined key types for asymmetric keys are:

- kSecAttrKeyTypeRSA
- kSecAttrKeyTypeECSECPrimeRandom

Please note that this assurance activity addresses both SFR components FCS_CKM.1(1)(MDF) and FCS_CKM.1(2)/WLAN defined in [ST] [\[1\]](#).

Test Activities

Assurance Activity AA-FCS_CKM.1-ATE-01

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

Key Generation for FIPS PUB 186-4 RSA Schemes

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

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

1. Random Primes:
 - Provable primes
 - Probable primes
2. Primes with Conditions:
 - Primes p_1 , p_2 , q_1 , q_2 , p and q shall all be provable primes
 - Primes p_1 , p_2 , q_1 , and q_2 shall be provable primes and p and q shall be probable primes
 - Primes p_1 , p_2 , q_1 , q_2 , p and q shall all be probable primes

To test the key generation method for the Random Provable primes method and for all the Primes with Conditions methods, the evaluator must seed the TSF key generation routine with sufficient data to deterministically generate the RSA key pair. This includes the random seed(s), the public exponent of the RSA key, and the desired key length. For each key length supported, the evaluator shall have the TSF generate 25 key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation.

If possible, the Random Probable primes method should also be verified against a known good implementation as described above. Otherwise, the evaluator shall have the TSF generate 10 keys pairs for each supported key length $nlen$ and verify:

- $n = p * q$
- p and q are probably prime according to Miller-Rabin tests

- $\text{GCD}(p-1, e) = 1$
- $\text{GCD}(q-1, e) = 1$
- $2^{16} < e < 2^{256}$ and e is an odd integer
- $|p-q| > 2^{(nlen/2 - 100)}$
- $p \geq \text{squareroot}(2) * (2^{(nlen/2 - 1)})$
- $q \geq \text{squareroot}(2) * (2^{(nlen/2 - 1)})$
- $2^{(nlen/2)} < d < \text{LCM}(p-1, q-1)$
- $e*d = 1 \bmod \text{LCM}(p-1, q-1)$

Summary

The evaluator performed CAVS testing of FIPS 186-4 RSA key generation by CoreCrypto in user space in accordance with the CAVP test procedures using the CAVS tool supplied to CAVP laboratories. The results have been validated by the CAVP. The CAVS certificates are listed in [ST] [section 8.2.3](#).

Assurance Activity AA-FCS_CKM.1-ATE-02

Key Generation for FIPS 186-4 Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test

For each supported NIST curve, i.e. P-256, P-384 and P-521, the evaluator shall require the implementation under test (IUT) to generate 10 private/public key pairs. The private key shall be generated using an approved random bit generator (RBG). To determine correctness, the evaluator shall submit the generated key pairs to the public key verification (PKV) function of a known good implementation.

FIPS 186-4 Public Key Verification (PKV) Test

For each supported NIST curve, i.e. P-256, P-384 and P-521, the evaluator shall generate 10 private/public key pairs using the key generation function of a known good implementation and modify five of the public key values so that they are incorrect, leaving five values unchanged (i.e. correct). The evaluator shall obtain in response a set of 10 PASS/FAIL values.

Summary

The evaluator performed CAVS testing of FIPS 186-4 ECDSA P-256 and P-384 key generation by CoreCrypto in user space in accordance with the CAVP test procedures using the CAVS tool supplied to CAVP laboratories. The results have been validated by the CAVP. The CAVS certificates are listed in [ST] [section 8.2.3](#).

Assurance Activity AA-FCS_CKM.1-ATE-03

Key Generation for Curve25519

The evaluator shall require the implementation under test (IUT) to generate 10 private/public key pairs. The private key shall be generated as specified in RFC 7748 using an approved random bit generator (RBG). To determine correctness, the evaluator shall submit the generated key pairs to the public key verification (PKV) function of a known good implementation.

Note: Assuming the PKV function of the good implementation will:

- confirm the private and public keys are 32-byte values
- confirm the three least significant bits of the most significant byte of the private key are zero
- confirm the most significant bit of the least significant byte is zero
- confirm the second most significant bit of the most significant byte is one
- calculate the expected public key from the private key and confirm it matches the supplied public key

The evaluator shall generate 10 private/public key pairs using the key generation function of a known good implementation and modify 5 of the public key values so that they are incorrect, leaving five values unchanged (i.e., correct). The evaluator shall obtain in response a set of 10 PASS/FAIL values.

Summary

The evaluator performed testing of Curve25519 key generation using a CAVS-like tool developed by atsec.

Assurance Activity AA-FCS_CKM.1-ATE-04

Key Generation for Finite-Field Cryptography (FFC)

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

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

Cryptographic and Field Primes:

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

and two ways to generate the cryptographic group generator g :

Cryptographic Group Generator:

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

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

Private Key:

- *$\text{len}(q)$ bit output of RBG where $1 \leq x \leq q-1$*
- *$\text{len}(q) + 64$ bit output of RBG, followed by a mod $q-1$ operation where $1 \leq x \leq q-1$*

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

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

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

- *$g \neq 0, 1$*
- *q divides $p-1$*
- *$g^q \bmod p = 1$*
- *$g^x \bmod p = y$*

for each FFC parameter set and key pair.

Summary

This SFR is not claimed in [ST] .

2.1.2.2 Extended: Cryptographic Key Support (REK) (FCS_CKM_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FCS_CKM_EXT.1-ASE-01

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

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



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

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

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

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

Summary

Section 8.1 in the [ST]  describes the *Hardware Protection Functions*. Section 8.2 in the [ST]  describes the *Cryptographic Support*.

Section 8.2.1 describes an overview of key management performed by the TOE. This description includes a description of a REK (the TOE UID) used by the TOE. Section 8.1.1 describes that the TOE UID (the REK) is stored in hardware-protected memory in the Secure Enclave and is not accessible to any other part of the system. The UID is not even known by Apple. When the device starts up, an ephemeral key is created, entangled with the UID which is used to encrypt the Secure Enclave's portion of the device's memory space. The UID is never passed to any other part of the system than the Secure Enclave. The REK stored in the Secure Enclave is not accessible to the "regular" processor. Even the software within the Secure Enclave cannot read the REK. It can only request encryption/decryption operations performed by a dedicated AES engine accessible only from the Secure Enclave. The UID is used to derive two other keys: the 0x89B and 0x835 keys. These keys are derived during boot by encrypting defined constants with the UID. These keys are used to wrap the EMF key, which is the file system master key, and the DKey, which is the device key. The AES Key Wrap algorithm is used to encrypt the 0x89B and 0x835 keys. The UID is not generated on the device, but rather at manufacturing time. It is generated in the production environment using a protected system with an RNG that complies with the requirements of NIST SP 800-90A (i.e., an Approved SP 800-90A DRBG implementation), as specified in FCS_RBG_EXT.1.1 and FCS_RBG_EXT.1.2.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.2.3 Cryptographic Key Establishment (FCS_CKM.2(1))

TSS Assurance Activities

Assurance Activity AA-FCS_CKM.2-1-ASE-01

The evaluator shall ensure that the supported key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme.

Summary

Section 6.2 in the [ST] [\[1\]](#) defines the *Cryptographic Support* SFRs implemented by the TOE. Section 8.2.1 in the [ST] [\[1\]](#) describes an *Overview of Key Management*.

The evaluator reviewed FCS_CKM.1.1(1), FCS_CKM.1.1(2), FCS_CKM.2(1) and FCS_CKM.2(2) and verified that the supported key establishment schemes in FCS_CKM.2.1 correspond to the key generation schemes identified in FCS_CKM.1.1. Two key establishment schemes are defined: RSA based and elliptic curve based. Section 8.2.1 describes how the TOE performs key management. RSA and elliptic curve based key establishment schemes can be used for EAP-TLS/TLS, Bluetooth, VPN or WLAN. Elliptic curve key establishment schemes are also used within the TOE key management.

Assurance Activity AA-FCS_CKM.2-1-ASE-02

SP800-56B Key Establishment Schemes

The evaluator shall verify that the TSS describes whether the TOE acts as a sender, a recipient, or both for RSA-based key establishment schemes.

Summary

Section 8.8 in the [ST] [\[1\]](#) describes the *Trusted Path/Channels (FTP)*.

The evaluator reviewed section 8.8 and confirms that the TOE can act both as the sender and the receiver for RSA-based key establishment schemes.

Assurance Activity AA-FCS_CKM.2-1-ASE-03

The evaluator shall ensure that the TSS describes how the TOE handles decryption errors. In accordance with NIST Special Publication 800-56B, the TOE must not reveal the particular error that occurred, either through the contents of any outputted or logged error message or through timing variations.

Summary

Section 8.2.2 in the [ST] [\[1\]](#) describes the *Storage of Persistent Secrets and Private Keys by the Agent*.

In the bulleted summary, it is explained that all decryption errors are handled in accordance with SP 800-56B.

Guidance Assurance Activities

Assurance Activity AA-FCS_CKM.2-1-AGD-01

The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key establishment scheme(s).

Summary

[ST] [\[1\]](#) section 6.2 *Cryptographic Support (FCS)* defines the TLS cipher suites supported by the TOE. No configuration method to alter these cipher suites is provided. Also, the TOE generates ECC curve 25519 keys when protecting files that may need to be updated when the device is locked as described in the section "NSFileProtectionCompleteUnlessOpen" in [ST] [\[1\]](#) section 8.2.1 *Overview of Key Management*. The 256-bit key size is defined by the curve and is not configurable.

The TOE generates RSA and NIST curve ECC keys when establishing a protected communications channel with TLS. The algorithm and key size used depends on the negotiation between the two endpoints. Cipher suites supported by the TOE are listed in [CCGUIDE] [\[1\]](#) Table 5 and Table 6.

[CCGUIDE] [\[1\]](#) section 3.2 *Crypto-Related Function Configuration* states that the TOE comes with two cryptographic modules that provide cryptographic support for the TOE: Apple iOS CoreCrypto Kernel Module v8 for ARM and Apple iOS CoreCrypto Module v8 for ARM.

Also, [CCGUIDE] [\[1\]](#) section 3.2.2 *Key Establishment* states that TOE performs the following key establishment:

- RSA-based scheme
- ECC-based scheme

[CCGUIDE] [\[1\]](#) Table 3 entry FCS_CKM.2(1) (MDF) states that no configuration is needed for key generation as the TOE provides a programming interface (API) that allows specification of the requested key sizes and key types.

Test Activities

Assurance Activity AA-FCS_CKM.2-1-ATE-01

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

Key Establishment Schemes

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

SP800-56A Key Establishment Schemes

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

Function Test

The Function test verifies the ability of the TOE to implement the key agreement schemes correctly. To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each supported key agreement scheme-key agreement role combination, KDF type, and, if supported, key confirmation role-key confirmation type combination, the tester shall generate 10 sets of test vectors. The data set consists of one set of domain parameter values (FFC) or the NIST approved curve (ECC) per 10 sets of public keys. These keys are static, ephemeral or both depending on the scheme being tested.

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

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

The evaluator shall verify the correctness of the TSF's implementation of a given scheme by using a known good implementation to calculate the shared secret value, derive the keying material DKM, and compare hashes or MAC tags generated from these values.

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

Validity Test

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

The evaluator shall inject an error in some of the test vectors to test that the TOE recognizes invalid key agreement results caused by the following fields being incorrect: the shared secret value Z, the DKM, the other information field OI, the data to be MACed, or the generated MACTag. If the TOE contains the full or partial (only ECC) public key validation, the evaluator will also individually inject errors in both parties' static public keys, both parties' ephemeral public keys and the TOE's static private key to assure the TOE detects errors in the public key validation function and/or the partial key validation function (in ECC only). At least two of the test vectors shall remain unmodified and therefore should result in valid key agreement results (they should pass).

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

SP800-56B Key Establishment Schemes

If the TOE acts as a sender, the following assurance activity shall be performed to ensure the proper operation of every TOE supported combination of RSA-based key establishment scheme:

To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each combination of supported key establishment scheme and its options (with or without key confirmation if supported, for each supported key confirmation MAC function if key confirmation is supported, and for each supported mask generation function if KTS-OAEP is supported), the tester shall generate 10 sets of test vectors. Each test vector shall include the RSA public key, the plaintext keying material, any additional input parameters if applicable, the MacKey and MacTag if key confirmation is incorporated, and the outputted ciphertext. For each test vector, the evaluator shall perform a key establishment encryption operation on the TOE with the same inputs (in cases where key confirmation is incorporated, the test shall use the MacKey from the test vector instead of the randomly generated MacKey used in normal operation) and ensure that the outputted ciphertext is equivalent to the ciphertext in the test vector.

If the TOE acts as a receiver, the following assurance activities shall be performed to ensure the proper operation of every TOE supported combination of RSA-based key establishment scheme:

To conduct this test the evaluator shall generate or obtain test vectors FCS_CKM.2.1(2) from a known good implementation of the TOE supported schemes. For each combination of supported key establishment scheme and its options (with or without key confirmation if supported, for each supported key confirmation MAC function if key confirmation is supported, and for each supported mask generation function if KTS-OAEP is supported), the tester shall generate 10 sets of test vectors. Each test vector shall include the RSA private key, the plaintext keying material (KeyData), any additional input parameters if applicable, the MacTag in cases where key confirmation is incorporated, and the outputted ciphertext. For each test vector, the evaluator shall perform the key establishment decryption operation on the TOE and ensure that the outputted plaintext keying material (KeyData) is equivalent to the plaintext keying material in the test vector. In cases where key confirmation is incorporated, the evaluator shall perform the key confirmation steps and ensure that the outputted MacTag is equivalent to the MacTag in the test vector.

The evaluator shall ensure that the TSS describes how the TOE handles decryption errors. In accordance with NIST Special Publication 800-56B, the TOE must not reveal the particular error that occurred, either through the contents of any outputted or logged error message or through timing variations. If KTS-OAEP is supported, the evaluator shall create separate contrived ciphertext values that trigger each of the three decryption error checks described in NIST Special Publication 800-56B section 7.2.2.3, ensure that each decryption attempt results in an error, and ensure that any outputted or logged error message is identical for each. If KTS-KEMKWS is supported, the evaluator shall create separate contrived ciphertext values that trigger each of the three decryption error checks described in NIST Special Publication 800-56B section 7.2.3.3, ensure that each decryption attempt results in an error, and ensure that any outputted or logged error message is identical for each.

Summary

The evaluator performed CAVS testing of ECDH primitive Z and RSA2 key establishment by CoreCrypto in user space in accordance with the CAVP test procedures using the CAVS tool supplied to CAVP laboratories. The results have been validated by the CAVP. The CAVS certificates are listed in [ST] section 8.2.3.

2.1.2.4 Cryptographic Key Establishment (While device is locked) (FCS_CKM.2(2))

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FCS_CKM.2-2-ATE-01

The test for SP800-56A and SP800-56B key establishment schemes is performed in association with FCS_CKM.2.1(1).

Curve22519 Key Establishment Schemes

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

Function Test

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

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

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

Validity Test

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

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

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

Summary

The evaluator performed testing of Curve25519 ECDH key establishment using a CAVS-like tool developed by atsec.

2.1.2.5 Extended: Cryptographic Key Random Generation (FCS_CKM_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FCS_CKM_EXT.2-ASE-01

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

Summary

Section 8.2.1 in the [ST]  describes the *Overview of Key Management*.

Every time a file is created, a new 256-bit AES key is generated by the hardware random number generator and post-processed by a SP 800-90A CTR-based DRBG. This key is used to encrypt the file it was generated for using AES-256-CBC, where the IV is calculated with the block offset into the file and then hashed with the SHA-1 algorithm.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.2.6 Extended: Cryptographic Key Generation (FCS_CKM_EXT.3)

TSS Assurance Activities

Assurance Activity AA-FCS_CKM_EXT.3-ASE-01

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

- *The evaluator shall review the TSS to verify that it contains a description of the PBKDF use to derive KEKs. This description must include the size and storage location of salts. This activity may be performed in combination with that for FCS_COP.1(5).*
- *If the symmetric KEK is generated by an RBG, the evaluator shall review the TSS to determine that it describes how the functionality described by FCS_RBG_EXT.1 is invoked. The evaluator uses the description of the RBG functionality in FCS_RBG_EXT.1 or documentation available for the operational environment to determine that the key size being requested is greater than or equal to the key size and mode to be used for the encryption/decryption of the data.*
- *If the KEK is generated according to an asymmetric key scheme, the evaluator shall review the TSS to determine that it describes how the functionality described by FCS_CKM.1 is invoked. The evaluator uses the description of the key generation functionality in FCS_CKM.1 or documentation available for the operational environment to determine that the key strength being requested is greater than or equal to 112 bits.*
- *If the KEK is formed from a combination, the evaluator shall verify that the TSS describes the method of combination and that this method is either an XOR, a KDF, or encryption.*

Summary

Section 8.2.1 in the [ST]  describes the *Overview of Key Management*.

The evaluator reviewed section 8.2.1 and verified that all keys used for data encryption are 256-bit AES keys. KEKs are also 256-bit AES keys. The passcode key is derived from the UID, the passcode and the Salt using the Concatenation Key Derivation Function (Approved Alternative 1) as described in 5.8.1 of NIST SP 800-56A. The KEKs are generated using an Approved NIST SP 800-90A as described in FCS_RBG_EXT.1.

Assurance Activity AA-FCS_CKM_EXT.3-ASE-02

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

Summary

A KDF is used, however the SP 800-56C KDF is used and not the SP 800-108 KDF. See AA-FCS_CKM_EXT.3-ASE-03

Assurance Activity AA-FCS_CKM_EXT.3-ASE-03

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

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

Summary

The evaluator notes that a KDF as defined in SP 800-56C is used, which is based on RFC5869. The KDF defined in the RFC complies with SP 800-56C, as far as the extraction and expansions steps. The RFC specifies the order of the concatenation. The implementation of the KDF uses HMAC-SHA-256.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FCS_CKM_EXT.3-ATE-01

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

Data Fields	Notations	
	SP 800-108	SP 800-56C
Pseudorandom function	PRF	PRF
Counter length	r	r
Length of output of PRF	h	h
Length of derived keying material	L	L
Length of input values	I_length	I_length
Pseudorandom input values I	K_1 (key derivation key)	Z (shared secret)
Pseudorandom salt values	n/a	s
Randomness extraction MAC	n/a	MAC

Table 1: Notations used in SP 800-108 and SP 800-56C

Counter Mode Tests:

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

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

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

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

Feedback Mode Tests:

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

- One or more pseudorandom functions that are supported by the implementation (PRF).
- The length (in bits) of the output of the PRF (h).
- Minimum and maximum values for the length (in bits) of the derived keying material (L). These values can be equal if only one value of L is supported. These must be evenly divisible by h .
- Up to two values of L that are NOT evenly divisible by h .
- Whether or not zero-length IVs are supported.

- Whether or not a counter is used, and if so:
 - One or more of the values {8, 16, 24, 32} that equal the length of the binary representation of the counter (r).
 - Location of the counter relative to fixed input data: before, after, or in the middle.
 - Counter before fixed input data: fixed input data string length (in bytes), fixed input data string value.
 - Counter after fixed input data: fixed input data string length (in bytes), fixed input data string value.
 - Counter in the middle of fixed input data: length of data before counter (in bytes), length of data after counter (in bytes), value of string input before counter, value of string input after counter.
 - The length (I_length) of the input values I .

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

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

Double Pipeline Iteration Mode Tests:

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

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

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

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

Summary

The evaluator performed a CAVS test of the KDF in the Secure Enclave Processor. The tester notes that TRRT #237 applies to this test.

2.1.2.7 Extended: Key Destruction (FCS_CKM_EXT.4)

TSS Assurance Activities

Assurance Activity AA-FCS_CKM_EXT.4-ASE-01

The evaluator shall check to ensure the TSS lists each type of plaintext key material (DEKs, software-based key storage, KEKs, trusted channel keys, passwords, etc.) and its generation and storage location.

The evaluator shall verify that the TSS describes when each type of key material is cleared (for example, on system power off, on wipe function, on disconnection of trusted channels, when no longer needed by the trusted channel per the protocol, when transitioning to the locked state, and possibly including immediately after use, while in the locked state, etc.).

The evaluator shall also verify that, for each type of key, the type of clearing procedure that is performed (cryptographic erase, overwrite with zeros, overwrite with random pattern, or block erase) is listed. If different types of memory are used to store the materials to be protected, the evaluator shall check to ensure that the TSS describes the clearing procedure in terms of the memory in which the data are stored.

Summary

Section 8.2.1 in the [ST] describes the *Overview of Key Management*.

Tables 6 and 7 in section 8.2.1 describes all the key material, purpose and storage location. Keys that are encrypted are not cleared unless the data they protect is erased. The DKey and the EMF key are stored wrapped, in block 0 of the flash memory (effaceable storage) and thus erasable very quickly if needed. The UID is stored unencrypted but is not accessible by any other part of the system than the Secure Enclave. All other keys are stored in volatile memory, wrapped. The TSS states that all clearing operations of keys stored in volatile memory are performed by Apple iOS CoreCrypto Kernel Module v8.0 for ARM and Apple iOS CoreCrypto Module v8.0 for ARM. The TSS specify that this clearing is performed using the `memset(0)` function to overwrite the memory containing keys and sensitive parameters.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FCS_CKM_EXT.4-ATE-01

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

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

For these tests the evaluator shall utilize appropriate development environment (e.g. a Virtual Machine) and development tools (debuggers, simulators, etc.) to test that keys are cleared, including all copies of the key that may have been created internally by the TOE during normal cryptographic processing with that key.

- **Test 1:** *Applied to each key held as plaintext in volatile memory and subject to destruction by overwrite by the TOE (whether or not the plaintext value is subsequently encrypted for storage in volatile or non-volatile memory). In the case where the only selection made for the destruction method key was removal of power, then this test is unnecessary. The evaluator shall:*
 1. *Record the value of the key in the TOE subject to clearing.*
 2. *Cause the TOE to perform a normal cryptographic processing with the key from Step #1.*
 3. *Cause the TOE to clear the key.*

4. Cause the TOE to stop the execution but not exit.
5. Cause the TOE to dump the entire memory of the TOE into a binary file.
6. Search the content of the binary file created in Step #5 for instances of the known key value from Step #1.
7. Break the key value from Step #1 into 3 similar sized pieces and perform a search using each piece.

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

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

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

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

Summary

For test 1, the evaluator is also lead tester for CoreCrypto FIPS 140-2 testing. Part of FIPS 140-2 testing includes zeroization. Since all software requiring cryptographic support uses CoreCrypto, the zeroization tests performed as in FIPS 140-2 fulfill the requirement of this Assurance Activity. This approach was approved in previous iOS evaluations.

For tests 2 and 3, the evaluator used kernel and secure enclave debuggers and simulators as well as test applications to check the zeroization operations for the Master Key and the class keys (A, B, C, D). He confirmed that all keys are no longer available after operations where they are expected to be destroyed.

2.1.2.8 Extended: TSF Wipe (FCS_CKM_EXT.5)

TSS Assurance Activities

Assurance Activity AA-FCS_CKM_EXT.5-ASE-01

The evaluator shall check to ensure the TSS describes how the device is wiped; and the type of clearing procedure that is performed (cryptographic erase or overwrite) and, if overwrite is performed, the overwrite procedure (overwrite with zeros, overwrite three or more times by a different alternating pattern, overwrite with random pattern, or block erase). If different types of memory are used to store the data to be protected, the evaluator shall check to ensure that the TSS describes the clearing procedure in terms of the memory in which the data are stored (for example, data stored on flash are cleared by overwriting once with zeros, while data stored on the internal persistent storage device are cleared by overwriting three times with a random pattern that is changed before each write).

Summary

Section 8.2.2 in the [ST] [\[1\]](#) describes the *Overview of Key Management*.

This section specifies that

"when a wipe command is issued, protected data is wiped by erasing the top level KEKs. Since all data-at-rest is encrypted with one of those keys, the device is wiped."

The overwriting of the KEK is done through a `memset(0)` operation that overwrites the memory (see previous assurance activity).

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FCS_CKM_EXT.5-ATE-01

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

The evaluator shall perform one of the following tests. The test before and after the wipe command shall be identical. This test shall be repeated for each type of memory used to store the data to be protected.

Method 1 for File-based Methods:

- **Test 1:** The evaluator shall enable encryption according to the AGD guidance. The evaluator shall create a user data (protected data or sensitive data) file, for example, by using an application. The evaluator shall use a tool provided by the developer to examine this data stored in memory (for example, by examining a decrypted files). The evaluator shall initiate the wipe command according to the AGD guidance provided for FMT_SMF_EXT.1. The evaluator shall use a tool provided by the developer to examine the same data location in memory to verify that the data has been wiped according to the method described in the TSS (for example, the files are still encrypted and cannot be accessed).

Method 2 for Volume-based Methods:

- **Test 1:** The evaluator shall enable encryption according to the AGD guidance. The evaluator shall create a unique data string, for example, by using an application. The evaluator shall use a tool provided by the developer to search decrypted data for the unique string. The evaluator shall initiate the wipe command according to the AGD guidance provided for FMT_SMF_EXT.1. The evaluator shall use a tool provided by the developer to search for the same unique string in decrypted memory to verify that the data has been wiped according to the method described in the TSS (for example, the files are still encrypted and cannot be accessed).

Summary

Reading effaceable memory is not possible. Debugging of the AppleKeyStore after a complete wipe of the device shows that new keys are generated by the Secure Enclave Processor and loaded into effaceable memory.

2.1.2.9 Extended: Salt Generation (FCS_CKM_EXT.6)

TSS Assurance Activities

Assurance Activity AA-FCS_CKM_EXT.6-ASE-01

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

Summary

Section 8.1 in the [ST] [describes the Hardware Protection Functions.](#)

The salt used for the password-based key derivation function PBKDF2 uses the Secure Enclave physical noise source, which uses AES with device UID as the key for the pseudo-random function. This salt is regenerated every time that the password changes. All other salts values uses in iOS are generated using the True Random Number Generator of the application processor, post-processed by a CTR-based SP 800-90A DRBG implementation.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.2.10 Extended: Cryptographic Key Support (REK) (FCS_CKM_EXT.7)

TSS Assurance Activities

Assurance Activity AA-FCS_CKM_EXT.7-ASE-01

The assurance activity for this element is performed in conjunction with the assurance activity for FCS_CKM_EXT.1.

Summary

Please refer to AA-FCS_CKM_EXT.1-ASE-01.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.2.11 Cryptographic operation (Confidentiality Algorithms) (FCS_COP.1(1))

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FCS_COP.1-1-ATE-01

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

AES-CBC Tests

- **Test 1: AES-CBC Known Answer Tests**

There are four Known Answer Tests (KATs), described below. In all KATs, the plaintext, ciphertext, and IV values shall be 128-bit blocks. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

- **Test 1.1:** KAT-1. To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from AES-CBC encryption of the given plaintext using a key value of all zeros and an IV of all zeros. Five plaintext values shall be encrypted with a 128-bit all-zeros key, and the other five shall be encrypted with a 256-bit all-zeros key.
To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input and AES-CBC decryption.
- **Test 1.2:** KAT-2. To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 key values and obtain the ciphertext value that results from AES-CBC encryption of an all-zeros plaintext using the given key value and an IV of all zeros. Five of the keys shall be 128-bit keys, and the other five shall be 256-bit keys.
To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using an all-zero ciphertext value as input and AES-CBC decryption.
- **Test 1.3:** KAT-3. To test the encrypt functionality of AES-CBC, the evaluator shall supply the two sets of key values described below and obtain the ciphertext value that results from AES encryption of an all-zeros plaintext using the given key value and an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second set shall have 256 256-bit keys. Key i in each set shall have the leftmost i bits be ones and the rightmost $N-i$ bits be zeros, for i in $[1, N]$.
To test the decrypt functionality of AES-CBC, the evaluator shall supply the two sets of key and ciphertext value pairs described below and obtain the plaintext value that results from AES-CBC decryption of the given ciphertext using the given key and an IV of all zeros. The first set of key/ciphertext pairs shall have 128 128-bit key/ciphertext pairs, and the second set of key/ciphertext pairs shall have 256 256-bit key/ciphertext pairs. Key i in each set shall have the leftmost i bits be ones and the rightmost $N-i$ bits be zeros, for i in $[1, N]$. The ciphertext value in each pair shall be the value that results in an all-zeros plaintext when decrypted with its corresponding key.
- **Test 1.4:** KAT-4. To test the encrypt functionality of AES-CBC, the evaluator shall supply the set of 128 plaintext values described below and obtain the two ciphertext values that result from AES-CBC encryption of the given plaintext using a 128-bit key value of all zeros with an IV of all zeros and using a 256-bit key value of all zeros with an IV of all zeros, respectively. Plaintext value i in each set shall have the leftmost i bits be ones and the rightmost $128-i$ bits be zeros, for i in $[1, 128]$.
To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using ciphertext values of the same form as the plaintext in the encrypt test as input and AES-CBC decryption.

- **Test 2: AES-CBC Multi-Block Message Test**

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

- **Test 3: AES-CBC Monte Carlo Tests**

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

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

$CT[i] = \text{AES-CBC-Encrypt}(\text{Key}, PT)$
 $PT = CT[i-1]$

The ciphertext computed in the 1000th iteration (i.e., $CT[1000]$) is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation. The evaluator shall test the decrypt functionality using the same test as for encrypt, exchanging CT and PT and replacing AES-CBC-Encrypt with AES-CBC-Decrypt.

AES-CCM Tests

- **Test 1:** The evaluator shall test the generation-encryption and decryption-verification functionality of AES-CCM for the following input parameter and tag lengths:

128 bit and 256 bit keys

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

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

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

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

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

- **Test 1.1:** For EACH supported key and associated data length and ANY supported payload, nonce and tag length, the evaluator shall supply one key value, one nonce value and 10 pairs of associated data and payload values and obtain the resulting ciphertext.
- **Test 1.2:** For EACH supported key and payload length and ANY supported associated data, nonce and tag length, the evaluator shall supply one key value, one nonce value and 10 pairs of associated data and payload values and obtain the resulting ciphertext.
- **Test 1.3:** For EACH supported key and nonce length and ANY supported associated data, payload and tag length, the evaluator shall supply one key value and 10 associated data, payload and nonce value 3-tuples and obtain the resulting ciphertext.
- **Test 1.4:** For EACH supported key and tag length and ANY supported associated data, payload and nonce length, the evaluator shall supply one key value, one nonce value and 10 pairs of associated data and payload values and obtain the resulting ciphertext.

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

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

AES-GCM Test

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

128 bit and 256 bit keys

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

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

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

- **Test 1:** The evaluator shall test the encrypt functionality using a set of 10 key, plaintext, AAD, and IV tuples for each combination of parameter lengths above and obtain the ciphertext value and tag that results from AES-GCM authenticated encrypt. Each supported tag length shall be tested at least once per set of 10. The IV value may be supplied by the evaluator or the implementation being tested, as long as it is known.

- **Test 2:** The evaluator shall test the decrypt functionality using a set of 10 key, ciphertext, tag, AAD, and IV 5-tuples for each combination of parameter lengths above and obtain a Pass/Fail result on authentication and the decrypted plaintext if Pass. The set shall include five tuples that Pass and five that Fail. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

XTS-AES Test

- **Test 1:** The evaluator shall test the encrypt functionality of XTS-AES for each combination of the following input parameter lengths:

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

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

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

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

- **Test 2:** The evaluator shall test the decrypt functionality of XTS-AES using the same test as for encrypt, replacing plaintext values with ciphertext values and XTS-AES encrypt with XTS-AES decrypt.

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

- **Test 1:** The evaluator shall test the authenticated encryption functionality of AES-KW for EACH combination of the following input parameter lengths:

128 and 256 bit key encryption keys (KEKs)

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

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

- **Test 2:** The evaluator shall test the authenticated-decryption functionality of AES-KW using the same test as for authenticated-encryption, replacing plaintext values with ciphertext values and AES-KW authenticated-encryption with AES-KW authenticated-decryption.
- **Test 3:** The evaluator shall test the authenticated-encryption functionality of AES-KWP using the same test as for AES-KW authenticated-encryption with the following change in the three plaintext lengths:
 - One plaintext length shall be one octet. One plaintext length shall be 20 octets (160 bits).
 - One plaintext length shall be the longest supported plaintext length less than or equal to 512 octets (4096 bits).
- **Test 4:** The evaluator shall test the authenticated-decryption functionality of AES-KWP using the same test as for AES-KWP authenticated-encryption, replacing plaintext values with ciphertext values and AES-KWP authenticated-encryption with AES-KWP authenticated-decryption.

Summary

The evaluator performed CAVS testing of

- AES-CBC
- AES-CCM
- AES-GCM
- AES-KW

by CoreCrypto in user space in accordance with the CAVP test procedures using the CAVS tool supplied to CAVP laboratories. The results have been validated by the CAVP. The CAVS certificates are listed in [ST] section 8.2.3.

XTS-AES testing was not performed because XTS-AES is not claimed in [ST].

2.1.2.12 Cryptographic operation (Hashing Algorithms) (FCS_COP.1(2))

TSS Assurance Activities

Assurance Activity AA-FCS_COP.1-2-ASE-01

The evaluator shall check that the association of the hash function with other TSF cryptographic functions (for example, the digital signature verification function) is documented in the TSS.

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

Summary

Section 8.2 in the [ST] describes the *Cryptographic Support*.

Tables 8 and 9 specify all the CAVS certificates received by the TOE, including the ones for the message digest algorithms. The CAVS certificate entry specifies whether the implementation has been tested in byte-oriented or bit-oriented mode and which hash functions have been tested. Specifically, SHS Certs. in row 5 of table 8 and row 11 of table 9 have been analyzed. The evaluator reviewed the CAVS certs and verified that the mode tested is byte-oriented messages. the hash functions also accept empty length messages.

Guidance Assurance Activities

Assurance Activity AA-FCS_COP.1-2-AGD-01

The evaluator checks the AGD documents to determine that any configuration that is required to be done to configure the functionality for the required hash sizes is present.

Summary

[CCGUIDE] section 3.2 *Crypto-Related Function Configuration* states that the TOE comes with two cryptographic modules that provide cryptographic support for the TOE: Apple iOS CoreCrypto Kernel Module v8 for ARM and Apple iOS CoreCrypto Module v8 for ARM.

[CCGUIDE] section 3.2.3 *Hashing* states that 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

[CCGUIDE] section 3.2.3 explicitly states:

"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."

This section of [CCGUIDE][\[1\]](#) also states that functions to perform hashing are provided as part of the Common Crypto library describes in the "Cryptographic Services Guide [CRYPTOGUIDE][\[2\]](#) and in detail in the "Common Crypto man pages" [CC-MAN][\[3\]](#). The functions that perform cryptographic hashing are:

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

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

Additional description is provided in [CC-MAN][\[3\]](#) including the following related functions:

- 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-256
- HMAC-SHA-384
- HMAC-SHA-512 with message digest sizes 160, 256, 384, 512 bits.

Test Activities

Assurance Activity AA-FCS_COP.1-2-ATE-01

The TSF may implement either bit-oriented or byte-oriented; both implementations are not required. The evaluator shall perform all of the following tests for each hash algorithm implemented by the TSF and used to satisfy the requirements of this PP.

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

- **Test 1: Short Messages Test: Bit-oriented Mode**
The evaluators devise an input set consisting of $m+1$ messages, where m is the block length of the hash algorithm. The length of the messages ranges sequentially from 0 to m bits. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.
- **Test 2: Short Messages Test: Byte-oriented Mode**
The evaluators devise an input set consisting of $m/8+1$ messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to $m/8$ bytes, with each message being an integral number of bytes. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.
- **Test 3: Selected Long Messages Test: Bit-oriented Mode**
*The evaluators devise an input set consisting of m messages, where m is the block length of the hash algorithm. The length of the i^{th} message is $512 + 99*i$, where $1 \leq i \leq m$. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*
- **Test 4: Selected Long Messages Test: Byte-oriented Mode**
*The evaluators devise an input set consisting of $m/8$ messages, where m is the block length of the hash algorithm. The length of the i^{th} message is $512 + 8*99*i$, where $1 \leq i \leq m/8$. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*

- **Test 5: Pseudorandomly Generated Messages Test**
This test is for byteoriented implementations only. The evaluators randomly generate a seed that is n bits long, where n is the length of the message digest produced by the hash function to be tested. The evaluators then formulate a set of 100 messages and associated digests by following the algorithm provided in Figure 1 of SHAVS. The evaluators then ensure that the correct result is produced when the messages are provided to the TSF.

Summary

The evaluator performed CAVS testing of

- SHA-1
- SHA-256
- SHA-384
- SHA-512

by CoreCrypto in user space in accordance with the CAVP test procedures using the CAVS tool supplied to CAVP laboratories. The results have been validated by the CAVP. The CAVS certificates are listed in [ST] [\[4\]](#) section 8.2.3.

2.1.2.13 Cryptographic operation (Signature Algorithms) (FCS_COP.1(3))

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FCS_COP.1-3-ATE-01

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

- **Test 1: ECDSA Algorithm Tests**
 - **Test 1.1: ECDSA FIPS 186-4 Signature Generation Test**
For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator shall generate 10 1024-bit long messages and obtain for each message a public key and the resulting signature values R and S . To determine correctness, the evaluator shall use the signature verification function of a known good implementation.
 - **Test 1.2: ECDSA FIPS 186-4 Signature Verification Test**
For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator shall generate a set of 10 1024-bit message, public key and signature tuples and modify one of the values (message, public key or signature) in five of the 10 tuples. The evaluator shall obtain in response a set of 10 PASS/FAIL values.
- **Test 2: RSA Signature Algorithm Tests**
 - **Test 2.1: Signature Generation Test**
The evaluator shall verify the implementation of RSA Signature Generation by the TOE using the Signature Generation Test. To conduct this test the evaluator must generate or obtain 10 messages from a trusted reference implementation for each modulus size/SHA combination supported by the TSF. The evaluator shall have the TOE use their private key and modulus value to sign these messages. The evaluator shall verify the correctness of the TSF's signature using a known good implementation and the associated public keys to verify the signatures.
 - **Test 2.2: Signature Verification Test**

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

Summary

The evaluator performed CAVS testing of

- RSA
- ECDSA

by CoreCrypto in user space in accordance with the CAVP test procedures using the CAVS tool supplied to CAVP laboratories. The results have been validated by the CAVP. The CAVS certificates are listed in [ST] [\[ST\]](#) section 8.2.3.

2.1.2.14 Cryptographic operation (Keyed Hash Algorithms) (FCS_COP.1(4))

TSS Assurance Activities

Assurance Activity AA-FCS_COP.1-4-ASE-01

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

Summary

Section 8.2 in the [ST] [\[ST\]](#) describes the *Cryptographic Support*.

Tables 8 and 9 specify all the CAVS certificates received by the TOE, including the ones for the HMAC algorithms. The CAVS certificate entry specifies the size of the HMAC key, output MAC length, hash function used and block size. The evaluator reviewed the CAVS Certs. received for the HMAC algorithm in table 8 and 9, and could find that the hash functions used are SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512; the key size is < block size, = block size and > block size. The SFR specifically defines the size of the keys, output size. The block size is defined by the HMAC and SHS standard.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FCS_COP.1-4-ATE-01

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

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

Summary

The evaluator performed CAVS testing of

- HMAC SHA-1
- HMAC SHA-256
- HMAC SHA-384
- HMAC SHA-512

by CoreCrypto in user space in accordance with the CAVP test procedures using the CAVS tool supplied to CAVP laboratories. The results have been validated by the CAVP. The CAVS certificates are listed in [ST] section 8.2.3.

2.1.2.15 Cryptographic operation (Password-based Key Derivation Functions) (FCS_COP.1(5))

TSS Assurance Activities

Assurance Activity AA-FCS_COP.1-5-ASE-01

The evaluator shall check that the TSS describes the method by which the password is first encoded and then fed to the SHA algorithm. The settings for the algorithm (padding, blocking, etc.) shall be described, and the evaluator shall verify that these are supported by the selections in this component as well as the selections concerning the hash function itself.

The evaluator shall verify that the TSS contains a description of how the output of the hash function is used to form the submask that will be input into the function and is the same length as the KEK as specified in FCS_CKM_EXT.3.

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

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

The evaluator shall verify that the iteration count for PBKDFs performed by the TOE comply with NIST SP 800-132 by ensuring that the TSS contains a description of the estimated time required to derive key material from passwords and how the TOE increases the computation time for password-based key derivation (including but not limited to increasing the iteration count)

Summary

Section 8.2.1.1 in the [ST] describes the *Password based key derivation*.

The PBKDF2 function is used to derive the passcode key from the passcode, the UID and the salt. The number of iterations is set to take at least 100 to 150 milliseconds (ms), which means that the number of iterations performed is dependent upon how many iterations can be performed within this time frame. The description of how the password is encoded and then fed to the SHA-256 algorithm for the PBKDF2 function, as described in NIST SP 800-132, option 2.b.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.2.16 Extended: HTTPS Protocol (FCS_HTTPS_EXT.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FCS_HTTPS_EXT.1-ATE-01

- **Test 1:** *The evaluator shall attempt to establish an HTTPS connection with a webserver, observe the traffic with a packet analyzer, and verify that the connection succeeds and that the traffic is identified as TLS or HTTPS. Other tests are performed in conjunction with FCS_TLSC_EXT.1. Certificate validity shall be tested in accordance with testing performed for FIA_X509_EXT.1, and the evaluator shall perform the following test:*
- **Test 2:** *The evaluator shall demonstrate that using a certificate without a valid certification path results in an application notification. Using the administrative guidance, the evaluator shall then load a certificate or certificates to the Trust Anchor Database needed to validate the certificate to be used in the function, and demonstrate that the function succeeds. The evaluator then shall delete one of the certificates, and show that the application is notified of the validation failure.*

Summary

The evaluator established a connection to a web server from the TOE using HTTPS and verified this by examining the s_server status page in the TOE web browser. He also verified that the warning about an untrusted certificate is displayed properly.

2.1.2.17 Extended: Initialization Vector Generation (FCS_IV_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FCS_IV_EXT.1-ASE-01

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

Summary

Section 8.2.1 in the [ST]  describes the *Overview of Key Management*.

The evaluator reviewed section 8.2.1 and found AES key wrapping according to RFC3394 is used for key wrapping. AES key wrapping according to RFC3394 does not required the formation of IVs.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

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

TSS Assurance Activities

Assurance Activity AA-FCS_RBG_EXT.1-ASE-01

Documentation shall be produced and the evaluator shall perform the activities in accordance with Appendix D, the "Clarification to the Entropy Documentation and Assessment".

The evaluator shall verify that the API documentation provided according to Section 5.2.2, includes the security functions described in FCS_RBG_EXT.1.3.

Summary

Section 2.1 in the [EAR] [\[1\]](#) addresses this assurance activity, as it describes the entropy source high level design, which includes the security functions described in FCS_RBG_EXT.1.3.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FCS_RBG_EXT.1-ATE-01

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

The evaluator shall perform the following tests.

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

If the RNG has prediction resistance enabled, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) generate a second block of random bits (4) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 - 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. "generate one block of random bits" means to generate random bits with number of returned bits equal to the Output Block Length (as defined in NIST SP800-90A).

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

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

Entropy input: the length of the entropy input value must equal the seed length.

Nonce: If a nonce is supported (CTR_DRBG with no Derivation Function does not use a nonce), the nonce bit length is one-half the seed length.

Personalization string: The length of the personalization string must be \leq seed length. If the implementation only supports one personalization string length, then the same length can be used for both values. If more than one string length is support, the evaluator shall use personalization strings of two different lengths. If the implementation does not use a personalization string, no value needs to be supplied.

Additional input: the additional input bit lengths have the same defaults and restrictions as the personalization string lengths.

Summary

The evaluator performed CAVS testing of

- CTR DRBG in CoreCrypto
- CTR DRBG in CPU hardware

in accordance with the CAVP test procedures using the CAVS tool supplied to CAVP laboratories. The results have been validated by the CAVP. The CAVS certificates are listed in [ST] [\[ST\]](#) section 8.2.3.

2.1.2.19 Extended: Cryptographic Algorithm Services (FCS_SRV_EXT.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-FCS_SRV_EXT.1-AGD-01

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

Summary

The evaluator examined the provided API documentation, namely [CRYPTOGUIDE] [\[CRYPTOGUIDE\]](#) and [CKTSREF] [\[CKTSREF\]](#) and verified that they includes the necessary description of the security functions/cryptographic algorithms.

Test Activities

Assurance Activity AA-FCS_SRV_EXT.1-ATE-01

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

Summary

All CAVS tests previously referenced use the standard API of CoreCrypto. The evaluator performed this test in the course of CAVS testing when loading the CAVS test tool with Xcode, executing it, and obtaining the expected result.

2.1.2.20 Extended: Cryptographic Key Storage (FCS_STG_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FCS_STG_EXT.1-ASE-01

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

Summary

Section 8.2.1 in the [ST] [\[ST\]](#) describes the *Overview of Key Management*. Section 8.2.2 describes the *Storage of Persistent Secrets and Private Keys by the Agent*.

The evaluator verified in FCS_STG_EXT.1 that the key storage is software-based for asymmetric, symmetric keys and persistent secrets. Section 8.2.1 describes how keys and data is stored and encrypted (wrapped) on iOS, and describes the different key classes. The evaluator reviewed figure 4 and determined that the TOE implements the required secure key storage.

Guidance Assurance Activities

Assurance Activity AA-FCS_STG_EXT.1-AGD-01

The evaluator shall review the AGD guidance to determine that it describes the steps needed to import or destroy keys/secrets. The evaluator shall also verify that the API documentation provided according to Section 5.2.2 includes the security functions (import, use, and destruction) described in these requirements. The API documentation shall include the method by which applications restrict access to their keys/secrets in order to meet FCS_STG_EXT.1.4.

Summary

[CCGUIDE] [\[1\]](#) section 3.2.6 *Keys/Secrets Import/Destruction* provides information about managing (i.e., import, use, destroy) of keys and secrets. It states that cryptographic keys are stored in keychains.

[CCGUIDE] [\[1\]](#) section 3.2.6 identifies the related API documentation for management of keys/secrets (i.e., import, use, destroy) which is provided in the *Managing Keys, Certificates, and Passwords* section of the "Cryptographic Services Guide" [CRYPTOGUIDE] [\[2\]](#), with the API specified in the "Certificate, Key, and Trust Services" [CKTSREF] [\[3\]](#) and the "Keychain Services Programming Guide" [KEYCHAINPG] [\[4\]](#), 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.

Test Activities

Assurance Activity AA-FCS_STG_EXT.1-ATE-01

The evaluator shall test the functionality of each security function:

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

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

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

- *For symmetric algorithms, the secret shall be used to encrypt data.*
- *For persistent secrets, the secret shall be compared to the imported secret.*

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

- *The evaluator shall deny the approvals to verify that the application is not able to use the key/secret as described.*
- *The evaluator shall repeat the test, allowing the approvals to verify that the application is able to use the key/secret as described.*

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

- **Test 3:** *The evaluator shall destroy keys/secrets of each supported type according to the AGD guidance. The evaluator shall write, or the developer shall provide access to, an application that destroys an imported key/secret.*

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

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

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

Summary

The evaluator successfully tested the import, use, and destruction of keys using custom code calling the CoreCrypto API and/or the Apple Configurator 2.

2.1.2.21 Extended: Encrypted Cryptographic Key Storage (FCS_STG_EXT.2)


FCS_STG_EXT.2.1

TSS Assurance Activities

Assurance Activity AA-FCS_STG_EXT.2.1-ASE-01

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

Summary

Section 8.2.1 in the [ST]  describes the *Overview of Key Management*.

The evaluator reviewed section 8.2.1 and specifically the diagram in figure 4 and determined that it describes the key hierarchy or all the key material used by iOS. This description defines the keys as 256-bit AES keys and explains how a new per-file key is generated, and how keys are wrapped and protected by the OS. The IV is determined on the memory location of the file (the IV is calculated with the block offset into the file).

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.


FCS_STG_EXT.2.2

TSS Assurance Activities

Assurance Activity AA-FCS_STG_EXT.2.2-ASE-02

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

Summary

Section 8.2.1 in the [ST]  describes the *Overview of Key Management*.

The evaluator reviewed section 8.2.1 and determined that every DEK is encrypted using AES key wrapping according to RFC3394 (AES Key Wrap) as defined in FCS_STG_EXT.2.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.


2.1.2.22 Extended: Integrity of Encrypted Key Storage (FCS_STG_EXT.3)

TSS Assurance Activities

Assurance Activity AA-FCS_STG_EXT.3-ASE-01

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

Summary

Section 8.2.1 in the [ST]  describes the *Overview of Key Management*.

The evaluator reviewed FCS_STG_EXT.3 and determined that the integrity mechanism used for each encrypted key is the one provided by the Key Wrap cipher mode defined in RFC3394 and NIST SP 800-38F. This mode includes the authentication as described in section 6.2 in SP 800-38F.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.2.23 Extended: TLS Client Protocol (FCS_TLSC_EXT.1)

FCS_TLSC_EXT.1.1

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.1-ASE-01

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

Summary

Section 8.8.1 *EAP-TLS and TLS* in the [ST] describes the EAP-TLS and TLS protocols and ciphersuites supported by the TOE.

Section 8.8.1 describes which protocols are supported by the TOE (TLS and EAP-TLS), and specifies how this protocols can be used for WLAN. The TOE supports EAP-TLS with TLS v1.0, v1.1 and v1.2 with the ciphersuites:

- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC5246
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC5246 (optional)
- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC5246 (optional)
- TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC5246 (optional)

The TOE supports TLS v1.2 with the ciphersuites:

- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 5246
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC5289 (optional)
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289 (optional)
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289 (optional)
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289 (optional)
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289 (optional)
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (can disabled in the OE)
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (can disabled in the OE)
- TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (can disabled in the OE)
- TLS_RSA_WITH_AES_128_GCM_SHA256 (can disabled in the OE)
- TLS_RSA_WITH_AES_256_CBC_SHA (can disabled in the OE)
- TLS_RSA_WITH_AES_128_CBC_SHA (can disabled in the OE)
- TLS_RSA_WITH_AES_256_GCM_SHA384 (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (can disabled in the OE)

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.1-AGD-01

The evaluator shall also check the operational guidance to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS.

Summary

[CCGUIDE] section 3.2.8 *TLS Configuration* provides information for configuring TLS. It states 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 used TLS cipher suites 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. It also states that TLS is provided by the APIs of iOS Security Framework, which uses the Apple iOS CoreCrypto Module v8 for ARM. The TOE's crypto module libraries implement TLS 1.2 (along with TLS 1.0 and TLS 1.1 for EAP-TLS) supporting the cipher suites listed in Table 6 (and Table 5 for EAP-TLS) of [CCGUIDE] which are duplicated below.

For TLS 1.2, the following cipher suites are supported by the TOE in the evaluated configuration:

- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- 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

Per [CCGUIDE] Table 5 in section 3.2.7 *EAP-TLS Configuration*, for TLS 1.0 and TLS 1.1 (EAP-TLS), the following cipher suites are supported by the TOE in the evaluated configuration:

- 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

Test Activities

Assurance Activity AA-FCS_TLSC_EXT.1.1-ATE-01

The evaluator shall write, or the ST author shall provide, an application for the purposes of testing TLS. The evaluator shall also perform the following tests:

- **Test 1:** *The evaluator shall establish a TLS connection using each of the ciphersuites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an EAP session. It is sufficient to observe the successful negotiation of a ciphersuite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic in an attempt to discern the ciphersuite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).*
- **Test 2:** *The evaluator shall attempt to establish the connection using a server with a server certificate that contains the Server Authentication purpose in the extendedKeyUsage field and verify that a connection is established. The evaluator will then verify that the client rejects an otherwise valid server certificate that lacks the Server Authentication purpose in the extendedKeyUsage field and a connection is not established. Ideally, the two certificates should be identical except for the extendedKeyUsage field.*
- **Test 3:** *The evaluator shall send a server certificate in the TLS connection that does not match the server-selected ciphersuite (for example, send a ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite or send a RSA certificate while using one of the ECDSA ciphersuites.) The evaluator shall verify that the TOE disconnects after receiving the server's Certificate handshake message.*
- **Test 4:** *The evaluator shall configure the server to select the TLS_NULL_WITH_NULL_NULL ciphersuite and verify that the client denies the connection.*
- **Test 5:** *The evaluator shall perform the following modifications to the traffic:*
 - *Change the TLS version selected by the server in the Server Hello to a non-supported TLS version (for example, 1.3 represented by the two bytes 03 04) and verify that the client rejects the connection.*
 - *Modify at least one byte in the server's nonce in the Server Hello handshake message, and verify that the client rejects the Server Key Exchange handshake message (if using a DHE or ECDHE ciphersuite) or that the server denies the client's Finished handshake message.*
 - *Modify the server's selected ciphersuite in the Server Hello handshake message to be a ciphersuite not presented in the Client Hello handshake message. The evaluator shall verify that the client rejects the connection after receiving the Server Hello.*

- (conditional) If an ECDHE or DHE ciphersuite is selected, modify the signature block in the Server's Key Exchange handshake message, and verify that the client rejects the connection after receiving the Server Key Exchange message.
- Modify a byte in the Server Finished handshake message, and verify that the client sends a fatal alert upon receipt and does not send any application data.
- Send a valid Server Finished message in plaintext and verify the client sends a fatal alert upon receipt and does not send any application data. The server's finished message shall contain valid verify_data and shall parse correctly using a network protocol analysis tool.

Summary

The evaluator verified each cipher by configuring the server to accept one at a time and testing them individually. The evaluator also tested certificates with valid and invalid extendedKeyUsage options and supported and unsupported ciphersuites selections. The evaluator modified OpenSSL to attempt a connection using the NULL cipher or one with a variety of other modifications to its traffic and verified that these connections were denied.

FCS_TLSC_EXT.1.2

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.2-ASE-01

The evaluator shall ensure that the TSS describes the client's method of establishing all reference identifiers from the application-configured reference identifier, including which types of reference identifiers are supported (e.g. Common Name, DNS Name, URI Name, Service Name, or other application-specific Subject Alternative Names) and whether IP addresses and wildcards are supported. The evaluator shall ensure that this description identifies whether and the manner in which certificate pinning is supported or used by the TOE.

Summary

Section 8.8.1 *EAP-TLS and TLS* in the [ST] describes the EAP-TLS and TLS protocols and ciphersuites supported by the TOE.

The evaluator reviewed section 8.8.1 and determined that the TOE will compare the DN contained within the certificate (the Subject CN, Subject Alternative Name fields, IP address or wildcards, if applicable) to the DN of the requested server. If the DN does not match, then the application requesting to establish a trusted channel cannot establish the connection.

Certificate pinning is supported by the TOE. The user of the TLS framework can use certificate pinning. Note that TLS clients in the TOE (such as Safari) do not support certificate pinning.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.2-AGD-01

The evaluator shall verify that the AGD guidance includes instructions for setting the reference identifier to be used for the purposes of certificate validation in TLS. In particular, the AGD guidance should describe the API used by applications for configuring the reference identifier.

Summary

[CCGUIDE] section 3.2.8.2 *Setting Reference Identifier* provides information for setting the reference identifier used for certification validation in TLS. The guidance/API documentation for setting the reference identifier is provided in the "Obtaining policies for establishing trust" section in the chapter 'Policies' in [CKTSREF].

Test Activities

Assurance Activity AA-FCS_TLSC_EXT.1.2-ATE-01

The evaluator shall configure the reference identifier according to the AGD guidance and perform the following tests during a TLS connection:

- **Test 1:** The evaluator shall present a server certificate that does not contain an identifier in either the Subject Alternative Name (SAN) or Common Name (CN) that matches the reference identifier. The evaluator shall verify that the connection fails.
- **Test 2:** The evaluator shall present a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported SAN type.
- **Test 3:** The evaluator shall present a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection succeeds.
- **Test 4:** The evaluator shall present a server certificate that contains a CN that does not match the reference identifier but does contain an identifier in the SAN that matches. The evaluator shall verify that the connection succeeds.
- **Test 5:** The evaluator shall perform the following wildcard tests with each supported type of reference identifier:
 - The evaluator shall present a server certificate containing a wildcard that is not in the left-most label of the presented identifier (e.g. foo.*.example.com) and verify that the connection fails.
 - The evaluator shall present a server certificate containing a wildcard in the left-most label but not preceding the public suffix (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection succeeds. The evaluator shall configure the reference identifier without a left-most label as in the certificate (e.g. example.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.example.com) and verify that the connection fails.
 - The evaluator shall present a server certificate containing a wildcard in the left-most label immediately preceding the public suffix (e.g. *.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.com) and verify that the connection fails.
- **Test 6:**[conditional] If URI or Service name reference identifiers are supported, the evaluator shall configure the DNS name and the service identifier. The evaluator shall present a server certificate containing the correct DNS name and service identifier in the URIName or SRVName fields of the SAN and verify that the connection succeeds. The evaluator shall repeat this test with the wrong service identifier (but correct DNS name) and verify that the connection fails.
- **Test 7:**[conditional] If pinned certificates are supported the evaluator shall present a certificate that does not match the pinned certificate and verify that the connection fails.

Summary

The evaluator set up an access point with a variety of server certificates created to reflect the condition specified in each of the tests and confirmed that the expected behavior occurred in each case.

FCS_TLSC_EXT.1.3

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FCS_TLSC_EXT.1.3-ATE-01

The evaluator shall perform the following test:

- **Test 1:** The evaluator shall demonstrate that using a certificate without a valid certification path results in the function failing. Using the administrative guidance, the evaluator shall then load a certificate or certificates to the Trust Anchor Database needed to validate the certificate to be used in the function, and demonstrate that the function succeeds. The evaluator then shall delete one of the certificates, and show that the function fails.

Summary

The evaluator created a two certificates, one using a valid CA and the other with a CA not trusted by TOE, and demonstrated the TOE does not complete the connection using the unknown CA certificate but does accept the connection using the valid CA certificate.

FCS_TLSC_EXT.1.4

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.4-ASE-01

The evaluator shall ensure that the TSS description required per FIA_X509_EXT.2.1 includes the use of client-side certificates for TLS mutual authentication.

Summary

Section 8.4.1 in the [ST] describes the certificates used by the TOE, and how iOS uses certificates for different services, including TLS and EAP-TLS authentication.

Section 8.4.2 describes that certificates can be installed for TLS mutual authentication.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.4-AGD-01

The evaluator shall verify that the AGD guidance required per FIA_X509_EXT.2.1 includes instructions for configuring the client-side certificates for TLS mutual authentication.

Summary

[CCGUIDE] sections 3.2.10 *Client Certificate Configuration* provides information for configuring client-side X.509 certificates which involves using a Configuration Profile as described in "Certificate Payload" of [IOS_CFG]. Additionally, [CCGUIDE] section 3.1 *Configuration Profile* provides additional detail about Configuration Profiles and [CCGUIDE] section 3.2.7 *EAP-TLS Configuration* provides details for configuring a WPA-EAP (when Wi-Fi Protected Access.)

Moreover, [CCGUIDE] section 3.4.7.1 *Certificate Validation* states that 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.

Test Activities

Assurance Activity AA-FCS_TLSC_EXT.1.4-ATE-01

The evaluator shall also perform the following test:

- **Test 1:** The evaluator shall perform the following modification to the traffic:
 - Configure the server to require mutual authentication and then modify a byte in a CA field in the Server's Certificate Request handshake message. The modified CA field must not be the CA used to sign the client's certificate. The evaluator shall verify the connection is unsuccessful.

Summary

The evaluator modified OpenSSL to alter traffic as described and verified that the connection then attempted was not successful.

2.1.2.24 Extended: TLS Protocol (FCS_TLSC_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.2.1-ASE-01

The evaluator shall verify that TSS describes the Supported Elliptic Curves Extension and whether the required behavior is performed by default or may be configured.

Summary

Section 8.8.1 in the ST specifies the *EAP-TLS* and *TLS* protocols. It is specified that the elliptic curve cipher suites may use the following elliptic curve extensions: secp256r1 (default), secp384r1 (default) and x25519 (may be disabled in the operational environment).

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.2.1-AGD-01

If the TSS indicates that the Supported Elliptic Curves Extension must be configured to meet the requirement, the evaluator shall verify that AGD guidance includes configuration of the Supported Elliptic Curves Extension.

Summary

Section 3.2.11 *Configuration of the Supported Elliptic Curves Extension* of [CCGUIDE][\[1\]](#) provides information for configuring elliptic curves extension for TLS. It states that the supported elliptic curves (i.e., secp256r1, secp384r1, x25519) are automatically selected by the TOE (i.e., the TOE does not support the individual selection of elliptic curves). 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.

Test Activities

Assurance Activity AA-FCS_TLSC_EXT.2.1-ATE-01

- **Test 1:** [TD0244] The evaluator shall configure a server to perform ECDHE key exchange using each of the TOE's supported curves and shall verify that the TOE successfully connects to the server.

Summary

The evaluator performed a test connection using a key generated for each of the TOE-supported curves and successfully established the connection each time.

2.1.2.25 Cryptographic Key Storage (FCS_STG_EXT.4(AGENT))

TSS Assurance Activities

Assurance Activity AA-FCS_STG_EXT.4-MDMA-MDM-ASE-01

The evaluator will verify that the TSS lists each persistent secret (credential, secret key) and private key needed to meet the requirements in the ST. For each of these items, the evaluator will confirm that the TSS lists for what purpose it is used, and, for each platform listed as supported in the ST, how it is stored. The evaluator shall verify that the Agent calls a platform-provided API to store persistent secrets and private keys.

Summary

Sections 8.2.1 and 8.2.2 of the [ST][\[1\]](#) describes the *Overview of Key Management and Storage of Persistent Secrets and Private Keys by the Agent*.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.2.26 Cryptographic Key Generation (FCS_CKM.1(2)(WLAN))

TSS Assurance Activities

Assurance Activity AA-FCS_CKM.1-WLAN-ASE-01

The evaluator shall verify that the TSS describes how the primitives defined and implemented by this EP are used by the TOE in establishing and maintaining secure connectivity to the wireless clients. The TSS shall also provide a description of the developer's method(s) of assuring that their implementation conforms to the cryptographic standards; this includes not only testing done by the developing organization, but also any third-party testing that is performed.

Summary

Section 8.8.4 *Wireless LAN* in the [ST][\[1\]](#) describes the WLAN protocol implemented by the TOE.

Section 8.8.4 describes that the WLAN protocol implements the protocol as defined in the IEEE 802.11 (2012). The TOE implements the AES-128-CTR block chaining mode along with the CBC-MAC authentication algorithm (CCMP protocol) required by IEEE 802.11 (2012). The TOE implements WPA2 Enterprise which received the following Wi-Fi Alliance certificates:

- WFA20103
- WFA20606
- WFA20607

- WFA55890
- WFA55891
- WFA55892
- WFA56011
- WFA56012

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FCS_CKM.1-WLAN-ATE-01

The evaluator shall perform the following tests:

- *Test 1: The evaluator shall configure the access point so the cryptoperiod of the session key is 1 hour. The evaluator shall successfully connect the TOE to the access point and maintain the connection for a length of time that is greater than the configured cryptoperiod. The evaluator shall use a packet capture tool to determine that after the configured cryptoperiod, a re-negotiation is initiated to establish a new session key. Finally, the evaluator shall determine that the renegotiation has been successful and the client continues communication with the access point.*
- *Test 2: The evaluator shall perform the following test using a packet sniffing tool to collect frames between the TOE and a wireless LAN access point:*
 - *Step 1: The evaluator shall configure the access point to an unused channel and configure the WLAN sniffer to sniff only on that channel (i.e., lock the sniffer on the selected channel). The sniffer should also be configured to filter on the MAC address of the TOE and/or access point.*
 - *Step 2: The evaluator shall configure the TOE to communicate with a WLAN access point using IEEE 802.11-2012 and a 256-bit (64 hex values 0-f) pre-shared key. The pre-shared key is only used for testing.*
 - *Step 3: The evaluator shall start the sniffing tool, initiate a connection between the TOE and the access point, and allow the TOE to authenticate, associate, and successfully complete the 4 way handshake with the client.*
 - *Step 4: The evaluator shall set a timer for 1 minute, at the end of which the evaluator shall disconnect the TOE from the wireless network and stop the sniffer.*
 - *Step 5: The evaluator shall identify the 4-way handshake frames (denoted EAPOL-key in Wireshark captures) and derive the PTK from the 4-way handshake frames and pre-shared key as specified in IEEE 802.11-2012.*
 - *Step 6: The evaluator shall select the first data frame from the captured packets that was sent between the TOE and access point after the 4-way handshake successfully completed, and without the frame control value 0x4208 (the first 2 bytes are 08 42). The evaluator shall use the PTK to decrypt the data portion of the packet as specified in IEEE 802.11-2012, and shall verify that the decrypted data contains ASCII-readable text.*
 - *Step 7: The evaluator shall repeat Step 6 for the next 2 data frames between the TOE and access point and without frame control value 0x4208.*

Summary

For test 1, the evaluator configured the TOE to renegotiate the WLAN connection after one hour, started Wireshark, and established the WLAN connection. After waiting for the hour to pass, the evaluator verified in the Wireshark traffic log that the renegotiation took place as expected.

For test 2, the evaluator then configured the TOE to wait for a WLAN connection using a pre-shared key. After the connection, the Wireshark traffic log was examined to check that the handshake was performed as described.

2.1.2.27 Cryptographic Key Distribution (GTK) (FCS_CKM.2(3)(WLAN))

TSS Assurance Activities

Assurance Activity AA-FCS_CKM.2-WLAN-ASE-01

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

Summary

Section 8.8.4 *Wireless LAN* in the [ST] describes the WLAN protocol implemented by the TOE. Paragraph 3 of this section specifically describes how the GTK is unwrapped as stated below.

The AES key wrapping algorithm is used and AES key unwrapping is performed as described in NIST SP 800-38F section 6.1 to unwrap the GTK, which is sent in an Extensible Authentication Protocol (EAPOL) key frame.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FCS_CKM.2-WLAN-ATE-01

The evaluator shall perform the following test using a packet sniffing tool to collect frames between the TOE and a wireless access point (which may be performed in conjunction with the assurance activity for FCS_CKM.1.1/WLAN).

Step 1: The evaluator shall configure the access point to an unused channel and configure the WLAN sniffer to sniff only on that channel (i.e., lock the sniffer on the selected channel). The sniffer should also be configured to filter on the MAC address of the TOE and/or access point.

Step 2: The evaluator shall configure the TOE to communicate with the access point using IEEE 802.11-2012 and a 256-bit (64 hex values 0-f) pre-shared key, setting up the connections as described in the operational guidance. The pre-shared key is only used for testing.

Step 3: The evaluator shall start the sniffing tool, initiate a connection between the TOE and access point, and allow the TOE to authenticate, associate, and successfully complete the 4-way handshake with the TOE.

Step 4: The evaluator shall set a timer for 1 minute, at the end of which the evaluator shall disconnect the TOE from the access point and stop the sniffer.

Step 5: The evaluator shall identify the 4-way handshake frames (denoted EAPOL-key in Wireshark captures) and derive the PTK and GTK from the 4-way handshake frames and pre-shared key as specified in IEEE 802.11-2012.

Step 6: The evaluator shall select the first data frame from the captured packets that was sent between the TOE and access point after the 4-way handshake successfully completed, and with the frame control value 0x4208 (the first 2 bytes are 08 42). The evaluator shall use the GTK to decrypt the data portion of the selected packet as specified in IEEE 802.11-2012, and shall verify that the decrypted data contains ASCII-readable text.

Step 7: The evaluator shall repeat Step 6 for the next 2 data frames with frame control value 0x4208.

Summary

The evaluator configured the TOE to wait for a WLAN connection using a pre-shared key. After the connection, the Wireshark traffic log was examined to check that the handshake was performed as described.


2.1.2.28 Extensible Authentication Protocol-Transport Layer Security (FCS_TLSC_EXT.1(WLAN))

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1-WLAN-ASE-01

The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the ciphersuites supported are specified. The evaluator shall check the TSS to ensure that the ciphersuites specified include those listed for this component. The evaluator shall also check the operational guidance to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS.

Summary

Section 8.8.1 EAP-TLS and TLS in the [ST]  describes the EAP-TLS and TLS protocols and ciphersuites supported by the TOE.

Section 8.8.1 describes which protocols are supported by the TOE (TLS and EAP-TLS), and specifies how this protocols can be used for WLAN. The TOE supports EAP-TLS with TLS v1.0, v1.1 and v1.2 with the ciphersuites:

- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC5246
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC5246 (optional)
- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC5246 (optional)
- TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC5246 (optional)

The TOE supports TLS v1.2 with the ciphersuites:

- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 5246
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC5289 (optional)
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289 (optional)
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289 (optional)
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289 (optional)
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289 (optional)
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (can disabled in the OE)
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (can disabled in the OE)
- TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (can disabled in the OE)
- TLS_RSA_WITH_AES_128_GCM_SHA256 (can disabled in the OE)
- TLS_RSA_WITH_AES_256_CBC_SHA (can disabled in the OE)
- TLS_RSA_WITH_AES_128_CBC_SHA (can disabled in the OE)
- TLS_RSA_WITH_AES_256_GCM_SHA384 (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (can disabled in the OE)
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (can disabled in the OE)

The evaluator checked section 3.2.8 of the [CCGUIDE] and could verify that there is no configuration required for the TOE: they are automatically selected by the TOE.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1-WLAN-AGD-01

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

The evaluator shall check that the OPE guidance contains instructions for the administrator to configure the list of Certificate Authorities that are allowed to sign certificates used by the authentication server that will be accepted by the TOE in the EAP-TLS exchange, and instructions on how to specify the algorithm suites that will be proposed and accepted by the TOE during the EAP-TLS exchange.

Summary

[CCGUIDE] section 3.2.7 *EAP-TLS Configuration* provides information for configuring EAP-TLS. It states that the supported ciphersuites listed in the Table 5 (reproduced below) 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 via a configuration profile as described in this section..

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 2: EAP-TLS Ciphersuites

[CCGUIDE] section 3.2.9 *Certificate Authority (CA) Configuration* provides information for configuring Certificate Authorities by stating that additional Certificate Authorities (beside those preinstalled with iOS) can be added using either the Apple Configurator or a Configuration Profile as described in [IOS_CFG].

Test Activities

Assurance Activity AA-FCS_TLSC_EXT.1-WLAN-ATE-01

The evaluator shall write, or the ST author shall provide, an application for the purposes of testing TLS.

The evaluator shall also perform the following tests:

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

- *Test 2: The evaluator shall attempt to establish the connection using a server with a server certificate that contains the Server Authentication purpose in the extendedKeyUsage field and verify that a connection is established. The evaluator will then verify that the client rejects an otherwise valid server certificate that lacks the Server Authentication purpose in the extendedKeyUsage field and a connection is not established. Ideally, the two certificates should be identical except for the extendedKeyUsage field.*
- *Test 3: The evaluator shall send a server certificate in the TLS connection that does not match the server-selected ciphersuite (for example, send a ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite or send a RSA certificate while using one of the ECDSA ciphersuites.) The evaluator shall verify that the TOE disconnects after receiving the server's Certificate handshake message.*
- *Test 4: The evaluator shall configure the server to select the TLS_NULL_WITH_NULL_NULL ciphersuite and verify that the client denies the connection.*
- *Test 5: The evaluator shall perform the following modifications to the traffic:*
 - *Change the TLS version selected by the server in the Server Hello to a non-supported TLS version (for example 1.3 represented by the two bytes 03 04) and verify that the client rejects the connection.*
 - *Modify at least one byte in the server's nonce in the Server Hello handshake message, and verify that the client rejects the Server Key Exchange handshake message (if using a DHE or ECDHE ciphersuite) or that the server denies the client's Finished handshake message.*
 - *Modify the server's selected ciphersuite in the Server Hello handshake message to be a ciphersuite not presented in the Client Hello handshake message. The evaluator shall verify that the client rejects the connection after receiving the Server Hello.*
 - *Modify the signature block in the Server's Key Exchange handshake message, and verify that the client rejects the connection after receiving the Server Key Exchange message.*
 - *Modify a byte in the Server Finished handshake message, and verify that the client sends a fatal alert upon receipt and does not send any application data.*
 - *Send a garbled message from the Server after the Server has issued the ChangeCipherSpec message and verify that the client denies the connection.*

Summary

The evaluator established TLS connections using each of the ciphersuites listed in [ST] as described in Test 1. He then created the certificates as specified in tests 2-4 to confirm that the connections were accepted or denied as expected.

For test 5, the evaluator used a modified version of OpenSSL allowing traffic to be modified as specified in the test. All tests resulted in the expected connection failure or error condition.

2.1.3 User data protection (FDP)

2.1.3.1 Extended: Security Access Control (FDP_ACF_EXT.1)

FDP_ACF_EXT.1.1

TSS Assurance Activities

Assurance Activity AA-FDP_ACF_EXT.1.1-ASE-01

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

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

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

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

For any services for which the user may grant access, the evaluator shall ensure that the TSS identifies whether the user is prompted for authorization when the application is installed, or during runtime.

Summary

Section 8.3 in the [ST] [\[1\]](#) describes the *User Data Protection (FDP)*.

Section 8.3 describes what an application is allowed to access on the system and which files it can access. Each application is sandboxed which means that it can only access the files that belong to this application. Regarding the capabilities of the application to access system services, every application must declare device-specific capabilities they need to run. These capabilities are defined in an array or dictionary that contain keys identifying features that the application requires. Applications owned by the same developed account can share content if configured to be part of an application group on the device. Creating such a appropriate group is up to the developer. Moreover, the TOE allows a user to restrict the applications to access certain system services such as:

- Location Services
- Contacts
- Calendar
- Reminders
- Photos
- Microphones
- Cameras

Guidance Assurance Activities

Assurance Activity AA-FDP_ACF_EXT.1.1-AGD-01

The evaluator shall ensure that the operational user guidance contains instructions for restricting application access to system services

Summary

[CCGUIDE] [\[1\]](#) section 3.3.3 *Restrict Application Access to System Services* states the following:

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] [\[1\]](#), chapter *iOS Keys*.

Test Activities

Assurance Activity AA-FDP_ACF_EXT.1.1-ATE-01

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

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

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

Summary

Tests 1 and 2 are not applicable because [ST] specifies no services that are not accessible to any application or that are only allowed access by privileged applications.

For test 3, the evaluator used two different editors to create files and verified that they could not access the data of the other. For both tests 3 and 4, the evaluator used two different browsers to access Keychain data and verified that one could not share data the other had imported, but both could access data intended to be accessed by all applications.

Testing of access control for other functions, such as cameras and microphones, is performed in association with the Assurance Activities of FMT_SMF_EXT.1.

FDP_ACF_EXT.1.2

TSS Assurance Activities

Assurance Activity AA-FDP_ACF_EXT.1.2-ASE-01

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

Summary

Section 8.3 in the [ST] describes the *User Data Protection (FDP)*.

Sections 8.3.1 and 8.3.2 describe that applications are sandboxed and can only access data within their own directory in the file tree structure, unless they are part of an app group set up by the application developer, as described in section 8.3.4. During the installation of the application, the installer creates app containers limited to the directories inside the application's sandbox which enforces which files the application can access.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FDP_ACF_EXT.1.2-ATE-01

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

Summary

The evaluator used calendar and email applications to show that data can be imported and exported between applications.

2.1.3.2 Extended: Protected Data Encryption (FDP_DAR_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDP_DAR_EXT.1-ASE-01

The evaluator shall verify that the TSS section of the ST indicates which data is protected by the DAR implementation and what data is considered TSF data. The evaluator shall ensure that this data includes all protected data.

Summary

Section 8.3.6 in the [ST]  describes *Keychain Data Protection*. Section 1.5.2.5 in the [ST]  describes the *Protection of the TSF*.


Section 8.3.6 describes how the data is separated in four different categories in table 12 . The security daemon determines which keychain items each process or application can access. Table 6 describes the keys and persistent secrets stored in iOS. Table 7 describes the keys and persistent storage stored by the Agent on iOS. In the summary after Figure 4, the evaluator verified that all file system items, file system metadata, files and key chains are encrypted and are considered TSF data protected as DAR.

Guidance Assurance Activities

Assurance Activity AA-FDP_DAR_EXT.1-AGD-01

The evaluator shall review the AGD guidance to determine that the description of the configuration and use of the DAR protection does not require the user to perform any actions beyond configuration and providing the authentication credential. The evaluator shall also review the AGD guidance to determine that the configuration does not require the user to identify encryption on a per-file basis.

Summary

[CCGUIDE]  section 3.3.1 *Data-At-Rest (DAR) Protection Configuration* provides information on setting up Data-At-Rest (DAR) protection. It states that data is always encrypted for protection. However, to ensure data protection, establishment of a passcode on the device is required. The effectiveness of the 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.

Test Activities

Assurance Activity AA-FDP_DAR_EXT.1-ATE-01

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

- **Test 1:** The evaluator shall enable encryption according to the AGD guidance. The evaluator shall create user data (non-system) either by creating a file or by using an application. The evaluator shall use a tool provided by the developer to verify that this data is encrypted when the product is powered off, in conjunction with Test 1 for FIA_UAU_EXT.1.

Summary

Bypass of encryption and decryption of TOE data is not possible. Encryption and decryption is performed directly by the storage controller and always uses at least the least restricted class D key. The TOE provides no way to directly access the raw stored data.

It is possible to test the key life cycle of the class A, B, C, and D keys, which is tested in FCS_CKM_EXT.4. This test approach was accepted in previous iOS evaluations.

2.1.3.3 Extended: Sensitive Data Encryption (FDP_DAR_EXT.2)

FDP_DAR_EXT.2.1

TSS Assurance Activities

Assurance Activity AA-FDP_DAR_EXT.2.1-ASE-01

The evaluator shall verify that the TSS includes a description of which data stored by the TSF (such as by native applications) is treated as sensitive. This data may include all or some user or enterprise data and must be specific regarding the level of protection of email, contacts, calendar appointments, messages, and documents.

The evaluator shall examine the TSS to determine that it describes the mechanism that is provided for applications to use to mark data and keys as sensitive. This description shall also contain information reflecting how data and keys marked in this manner are distinguished from data and keys that are not (for instance, tagging, segregation in a "special" area of memory or container, etc.).

Summary

Section 8.2.1 of the [ST]  describes an *Overview of Key Management*.

Section 8.2.1 describes how key management works on iOS. iOS treats all data equally and encrypts all data the same way: every time a file is created, a new AES key is generated and used to encrypt this file in memory. This AES key is then wrapped by a group key.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FDP_DAR_EXT.2.1-ATE-01

- **Test 1:** The evaluator shall enable encryption of sensitive data and require user authentication according to the AGD guidance. The evaluator shall try to access and create sensitive data (as defined in the ST and either by creating a file or using an application to generate sensitive data) in order to verify that no other user interaction is required.

Summary

The storage encryption outlined for FDP_DAR_EXT.1 is the same that provides the extended storage encryption support by using the class keys. The calling application can select the respective class key to be used to wrap the session key used to protect the data at rest. Thus, the testing of FDP_DAR_EXT.1 covers the data storage part of the test requirement. In addition, the key life cycle discussion provided in FCS_CKM_EXT.4 demonstrates the key management side of the test requirement. Therefore, both analyses fully cover the test requirement of this SFR.


FDP_DAR_EXT.2.2

TSS Assurance Activities

Assurance Activity AA-FDP_DAR_EXT.2.2-ASE-01

The evaluator shall review the TSS section of the ST to determine that the TSS includes a description of the process of receiving sensitive data while the device is in a locked state. The evaluator shall also verify that the description indicates if sensitive data that may be received in the locked state is treated differently than sensitive data that cannot be received in the locked state. The description shall include the key scheme for encrypting and storing the received data, which must involve an asymmetric key and must prevent the sensitive data-at-rest from being decrypted by wiping all key material used to derive or encrypt the data (as described in the application note). The introduction to this section provides two different schemes that meet the requirements, but other solutions may address this requirement.

Summary

Section 8.2.1 in the [ST]  describes the *Overview of Key Management*.

The paragraph about the NSFileProtectionCompleteUnlessOpen describes how files can be retrieved on iOS although the device is locked, for example downloading an email attachment while the device is the locked state. This is achieved using asymmetric cryptography and elliptic curves. iOS specifically uses curve 25519: iOS generates a device-wide asymmetric key for the NSFileProtectionCompleteUnlessOpen file class within the Secure Enclave. When data is received by iOS in the locked state, iOS generates another asymmetric key pair. The device wide public key and the file object private key are used to generate a shared secret (one-pass Diffie-Hellman), which is post-processed by a KDF to obtain a symmetric key. This key is used to encrypt the data that is being received. Then, the file object private key is destroyed with the shared secret when the file is closed and only the public key is stored with the file. To access the file, the user has to provide the correct passcode to unwrap the device-wide private key. The files that have been encrypted with the above scheme have their key re-wrapped by the NSFileProtectionCompleetUnlessOpenclass key. It is up to the application to check whether the device is locked or not and to re-wrap the file with the correct key.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FDP_DAR_EXT.2.2-ATE-01

The evaluator shall perform the tests in FCS_CKM_EXT.4 for all key material no longer needed while in the locked state and shall ensure that keys for the asymmetric scheme are addressed in the tests performed when transitioning to the locked state.

Summary

See the key life cycle testing performed as part of FCS_CKM_EXT.4 which covers the destruction of all class keys, including the private and public class B key referenced in this SFR.

FDP_DAR_EXT.2.3


TSS Assurance Activities

Assurance Activity AA-FDP_DAR_EXT.2.3-ASE-01

The evaluator shall verify that the key hierarchy section of the TSS required for FCS_STG_EXT.2.1 includes the symmetric encryption keys (DEKs) used to encrypt sensitive data. The evaluator shall ensure that these DEKs are encrypted by a key encrypted with (or chain to a KEK encrypted with) the REK and password-derived or biometric-unlocked KEK.

The evaluator shall verify that the TSS section of the ST that describes the asymmetric key scheme includes the protection of any private keys of the asymmetric pairs. The evaluator shall ensure that any private keys that are not wiped and are stored by the TSF are stored encrypted by a key encrypted with (or chain to a KEK encrypted with) the REK and password-derived or biometric-unlocked KEK.

Summary

Section 8.2.1 in the [ST]  describes the *Overview of Key Management*.

Section 8.2.1 and specifically figure 4 describe how key hierarchy is implemented within iOS. The evaluator verified that the DEKs are encrypted by a key that is chained to the REK. Some keys, like the passcode is actually password derived (using the REK as well). Regarding asymmetric cryptography keys, the private keys are wrapped whenever the device is locked. These keys will be unwrapped when the user provides the correct passcode for the device.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

FDP_DAR_EXT.2.4

TSS Assurance Activities

Assurance Activity AA-FDP_DAR_EXT.2.4-ASE-01

The evaluator shall verify that the TSS section of the ST that describes the asymmetric key scheme includes a description of the actions taken by the TSF for the purposes of DAR upon transitioning to the unlocked state. These actions shall minimally include decrypting all received data using the asymmetric key scheme and re-encrypting with the symmetric key scheme used to store data while the device is unlocked.

Summary

Section 8.2.1 in the [ST]  describes the *Overview of Key Management*.

Asymmetric cryptography is used for verifying signatures of the software, protocols (key establishment), and when the device receives data while it is locked. The private keys are wrapped by symmetric cryptography whenever the device is locked. The symmetric used to wrap these keys

is destroyed and can only be re-generated whenever the user provides the correct passcode. When the device is unlocked, the keys protecting data received during the locked state are being unwrapped and re-wrapped with a different key.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.3.4 Extended: Subset Information Flow Control (FDP_IFC_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDP_IFC_EXT.1-ASE-01

The evaluator shall verify that the TSS section of the ST describes the routing of IP traffic through processes on the TSF when a VPN client is enabled. The evaluator shall ensure that the description indicates which traffic does not go through the VPN and which traffic does and that a configuration exists for each baseband protocol in which only the traffic identified by the ST author as necessary for establishing the VPN connection (IKE traffic and perhaps HTTPS or DNS traffic) is not encapsulated by the VPN protocol (IPsec). The evaluator shall verify that the TSS section describes any differences in the routing of IP traffic when using any supported baseband protocols (e.g. Wi-Fi or, LTE).

Summary

This aspect of the TOE is taken care of by another on-going evaluation performed by another Common Criteria laboratory.

Guidance Assurance Activities

Assurance Activity AA-FDP_IFC_EXT.1-AGD-01

The evaluator shall verify that one (or more) of the following options is addressed by the documentation:

- *The description above indicates that if a VPN client is enabled, all configurations route all Data Plane traffic through the tunnel interface established by the VPN client*
- *The AGD guidance describes how the user and/or administrator can configure the TSF to meet this requirement.*
- *The API documentation includes a security function that allows a VPN client to specify this routing.*

Summary

[CCGUIDE][\[1\]](#) section 3.3.2 *VPN/Wi-Fi Configuration* provides information to configure VPN connections. It states that 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 shown in the sample VPN Payload in section 3.1 *Configuration Profile* of [CCGUIDE][\[1\]](#).

Test Activities

Assurance Activity AA-FDP_IFC_EXT.1-ATE-01

- **Test 1:** *If the ST author identifies any differences in the routing between Wi-Fi and cellular protocols, the evaluator shall repeat this test with a base station implementing one of the identified cellular protocols.*

- *Step 1: The evaluator shall enable a Wi-Fi configuration as described in the AGD guidance (as required by FTP_ITC_EXT.1). The evaluator shall use a packet sniffing tool between the wireless access point and an Internet-connected network. The evaluator shall turn on the sniffing tool and perform actions with the device such as navigating to websites, using provided applications, and accessing other Internet resources. The evaluator shall verify that the sniffing tool captures the traffic generated by these actions, turn off the sniffing tool, and save the session data.*
- *Step 2: The evaluator shall configure an IPsec VPN client that supports the routing specified in this requirement, and if necessary, configure the device to perform the routing specified as described in the AGD guidance. The evaluator shall turn on the sniffing tool, establish the VPN connection, and perform the same actions with the device as performed in the first step. The evaluator shall verify that the sniffing tool captures traffic generated by these actions, turn off the sniffing tool, and save the session data.*
- *Step 3: The evaluator shall examine the traffic from both step one and step two to verify that all Data Plane traffic is encapsulated by IPsec. The evaluator shall examine the Security Parameter Index (SPI) value present in the encapsulated packets captured in Step two from the TOE to the Gateway and shall verify this value is the same for all actions used to generate traffic through the VPN. Note that it is expected that the SPI value for packets from the Gateway to the TOE is different than the SPI value for packets from the TOE to the Gateway. The evaluator shall be aware that IP traffic on the cellular baseband outside of the IPsec tunnel may be emanating from the baseband processor and shall verify with the manufacturer that any identified traffic is not emanating from the application processor.*
- *Step 4: The evaluator shall perform an ICMP echo from the TOE to the IP address of another device on the local wireless network and shall verify that no packets are sent using the sniffing tool. The evaluator shall attempt to send packets to the TOE outside the VPN tunnel (i.e. not through the VPN gateway), including from the local wireless network, and shall verify that the TOE discards them.*

Summary

The evaluator configured the TOE to use Wi-Fi in the clear as described by [CCGUIDE] and gathered traffic data. Then he configured the TOE to use IPsec and gathered traffic data for that connection. He then examined the IPsec traffic and confirmed that the HTTP traffic is encapsulated by ESP and verified the SPI for both directions.

The cellular network test is implicit. The evaluator used a debug session with the development team to verify that the VPN tunnel is always used (always-on VPN) or based on the IP stack routing table. The routing decision is made above the level of the network interface. The Wi-Fi and Cellular interfaces are standard network interfaces as viewed by the IP stack.

2.1.3.5 Extended: Storage of Critical Biometric Parameters (FDP_PBA_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDP_PBA_EXT.1-ASE-01

The evaluator shall determine that the TSS contains a description of the activities that happen during biometric authentication.

Summary

Section 8.4.1 in the [ST] specifies what happens when a user tries to authenticate to a device using biometrics. Biometric authentication do not produce feedback unless the input is rejected: in that case, when an invalid fingerprint is provided or can't be authenticated, an error message is displayed to the user. For facial recognition, the device will vibrate; when three invalid biometric samples are presented the device will require password/passcode authentication.

Guidance Assurance Activities



No assurance activities defined.

Test Activities

Assurance Activity AA-FDP_PBA_EXT.1-ATE-01

The evaluator shall ensure that the authentication template is protected either using a PIN or by other secure means, as specified by the vendor.

Summary

[CCGUIDE]  section 3.4.1 states the user must enable the use of password/passcode in the evaluated configuration. The user also has the option of using a biometric authentication mechanism, Touch ID or Face ID depending on platform model, in addition to the passcode. This is described in [CCGUIDE]  section 3.4.2.

2.1.3.6 Extended: User Data Storage (FDP_STG_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDP_STG_EXT.1-ASE-01

The evaluator shall ensure the TSS describes the Trust Anchor Database implemented that contain certificates used to meet the requirements of this PP. This description shall contain information pertaining to how certificates are loaded into the store, and how the store is protected from unauthorized access (for example, UNIX permissions) in accordance with the permissions established in FMT_SMF_EXT.1 and FMT_MOF_EXT.1.1.

Summary

Section 8.4.2 in the [ST]  describes how certificates are used in iOS.

The Apple certificate is installed in ROM during manufacturing. Other certificates can be imported by a user (if allowed by the policy) or installed using configuration profiles. The trust anchor database is the database with the CA roots used in TLS and the other protocols. That is hard-coded in the code and cannot be changed. The database is protected the same way as the binary code of the TOE, i.e., by the TOE integrity check and write protection.

Every certificate has a certificate type which defines what the certificate is used for. This ensured that only certificates defined for a specific application can be used for this application, and nothing else.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.3.7 1 Extended: Inter-TSF User Data Transfer Protection (FDP_UPC_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDP_UPC_EXT.1-ASE-01

The evaluator shall examine the TSS to determine that it describes that all protocols listed in the TSS are specified and included in the requirements in the ST.

Summary

Section 8.8 in the [ST] [\[ST\]](#) describes the *Trusted Path/Channels (FTP)*.

Section 8.8 is divided into several sections where each lists the protocol supported by the TOE: EAP-TLS, TLS, IPsec, Bluetooth and WLAN. Note that IPsec is addressed in a separate evaluation, as stated in section 1.5.2.7 of the [ST] [\[ST\]](#). These protocols are specified and included in the requirements in the [ST] [\[ST\]](#).

Guidance Assurance Activities

Assurance Activity AA-FDP_UPC_EXT.1-AGD-01

The evaluator shall verify that the API documentation provided according to Section 5.2.2 includes the security functions (protection channel) described in these requirements, and verify that the APIs implemented to support this requirement include the appropriate settings/parameters so that the application can both provide and obtain the information needed to assure mutual identification of the endpoints of the communication as required by this component. The evaluator shall write, or the developer shall provide access to, an application that requests protected channel services by the TSF. The evaluator shall verify that the results from the protected channel match the expected results according to the API documentation. This application may be used to assist in verifying the protected channel assurance activities for the protocol requirements.

Summary

[CCGUIDE] [\[CCGUIDE\]](#) section 3.2.8.1 *Setting Up TLS/HTTPS Channel* provides references to API documentation for setting up TLS/HTTPS channel. The guidance is provided in:

- Security Framework Reference [SECFWREF] [\[SECFWREF\]](#) in *Secure Transport Reference*.
- "Technical Note TN 2232: HTTPS Server Trust Evaluation [HTTPSTN2232] [\[HTTPSTN2232\]](#) for additional HTTPS guidance

Assurance Activity AA-FDP_UPC_EXT.1-AGD-02

The evaluator shall confirm that the operational guidance contains instructions necessary for configuring the protocol(s) selected for use by the applications.

Summary

[CCGUIDE] [\[CCGUIDE\]](#) section 3.2.8.1 *Setting Up TLS/HTTPS Channel* provides references to API documentation for setting up TLS/HTTPS channel. The guidance is provided in:

- Security Framework Reference [SECFWREF] [\[SECFWREF\]](#) in *Secure Transport Reference*.
- "Technical Note TN 2232: HTTPS Server Trust Evaluation [HTTPSTN2232] [\[HTTPSTN2232\]](#) for additional HTTPS guidance

Test Activities

Assurance Activity AA-FDP_UPC_EXT.1-ATE-01

[The evaluator shall write, or the developer shall provide access to, an application that requests protected channel services by the TSF. The evaluator shall verify that the results from the protected channel match the expected results according to the API documentation. This application may be used to assist in verifying the protected channel assurance activities for the protocol requirements.]

Summary

Protected services over TLS, HTTPS, and VPN are tested by Assurance Activities for FCS_TLCS_EXT.1, FCS_TLSC_EXT.2, and FDP_ITC_EXT.1.

The evaluator set up Bluetooth, Bluetooth BR/EDR, and Bluetooth LE connections between the TOE and another system and verified that link encryption was active both by checking configuration options and by sniffing the traffic generated after pairing the devices.

Assurance Activity AA-FDP_UPC_EXT.1-ATE-02

The evaluator shall also perform the following tests:

- **Test 1:** *The evaluators shall ensure that the application is able to initiate communications with an external IT entity using each protocol specified in the requirement, setting up the connections as described in the operational guidance and ensuring that communication is successful.*
- **Test 2:** *The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data are not sent in plaintext.*

Summary

Protected services over TLS, HTTPS, and VPN are tested by Assurance Activities for FCS_TLCS_EXT.1, FCS_TLSC_EXT.2, and FDP_ITC_EXT.1.

The evaluator set up Bluetooth, Bluetooth BR/EDR, and Bluetooth LE connections between the TOE and another system and verified that link encryption was active both by checking configuration options and by sniffing the traffic generated after pairing the devices.

2.1.4 Identification and authentication (FIA)

2.1.4.1 Extended: Authentication Failure Handling (FIA_AFL_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FIA_AFL_EXT.1-ASE-01

The evaluator shall ensure that the TSS describes that a value corresponding to the number of unsuccessful authentication attempts since the last successful authentication is kept for each Authentication Factor interface. The evaluator shall ensure that this description also includes if and how this value is maintained when the TOE loses power, either through a graceful powered off or an ungraceful loss of power. The evaluator shall ensure that if the value is not maintained, the interface is after another interface in the boot sequence for which the value is maintained.

Summary

Section 8.4 in the [ST] [\[1\]](#) describes the *Identification and Authentication (FIA)*.

A user can authenticate to the TOE using a passcode. The value of the number of unsuccessful authentication attempts since the last authentication is kept in a system file which persists in the event of a graceful or ungraceful loss of power to the TOE. This value can be set by the administrator of the device.

Assurance Activity AA-FIA_AFL_EXT.1-ASE-02

If the TOE supports multiple authentication mechanisms, the evaluator shall ensure that this description also includes how the unsuccessful authentication attempts for each mechanism selected in FIA_UAU.5.1 is handled. The evaluator shall verify that the TSS describes if each authentication mechanism utilizes its own counter or if multiple authentication mechanisms utilize a shared counter. If multiple authentication mechanisms utilize a shared counter, the evaluator shall verify that the TSS describes this interaction.

Summary

Section 8.4 of the [ST] describes the *Identification and Authentication (FIA)*.

iOS supports three different authentication mode: passcode, Face ID or Touch ID claimed for the TOE. The passcode and the biometric method on the phone share different unsuccessful attempt counters. Section 8.4.1 describes the Face ID and Touch ID authentication mechanism, and section 8.4 describes the general authentication rules on iOS. The passcode counter is stored in a file on the file system, and as soon as iOS triggers an authentication failure, the counter is updated, which prevents an attacker to power-off the phone before the counter is updated.

Assurance Activity AA-FIA_AFL_EXT.1-ASE-03

The evaluator shall confirm that the TSS describes how the process used to determine if the authentication attempt was successful. The evaluator shall ensure that the counter would be updated even if power to the device is cut immediately following notifying the TOE user if the authentication attempt was successful or not.

Summary

Section 8.4 specifies that

"The number of failed authentication attempts is maintained in a system file which will persist in the event of graceful or ungraceful loss of power to the TOE. The counter maintaining the number of failed consecutive logon attempts is increased by one immediately once the TOE has identified that the passcode is incorrect. The increment of the counter is completed before the UI informs the user about the failed logon attempt."

Regarding the unsuccessful attempts, section 8.4 also specifies:

"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."

Guidance Assurance Activities

Assurance Activity AA-FIA_AFL_EXT.1-AGD-01

The evaluator shall verify that the AGD guidance describes how the administrator configures the maximum number of unique unsuccessful authentication attempts.

Summary

[CCGUIDE] section 3.4.3 *Authentication Attempt Configuration* provides instructions for setting the authentication attempt. It states the following:

- To limit/configure the number of consecutive failed authentication attempts for the passcode; the administrator can use a Configuration Profile. 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.

Test Activities

Assurance Activity AA-FIA_AFL_EXT.1-ATE-01

- **Test 1:** The evaluator shall configure the device with all authentication mechanisms selected in FIA_UAU.5.1. The evaluator shall perform the following tests for each available authentication interface:
 - **Test 1a:** The evaluator shall configure the TOE, according to the AGD guidance, with a maximum number of unsuccessful authentication attempts. The evaluator shall enter the locked state and enter incorrect passwords until the wipe occurs. The evaluator shall verify that the number of password entries corresponds to the configured maximum and that the wipe is implemented.
 - **Test 1b:**[conditional] If the TOE supports multiple authentication mechanisms the previous test shall be repeated using a combination of authentication mechanisms confirming that the critical authentication mechanisms will cause the device to wipe and that when the maximum number of unsuccessful authentication attempts for a non-critical authentication mechanism is exceeded, the device limits authentication attempts to other available authentication mechanisms. If multiple authentication mechanisms utilize a shared counter, then the evaluator shall verify that the maximum number of unsuccessful authentication attempts can be reached by using each individual authentication mechanism and a combination of all authentication mechanisms that share the counter.
- **Test 2:** The evaluator shall repeat test one, but shall power off (by removing the battery, if possible) the TOE between unsuccessful authentication attempts. The evaluator shall verify that the total number of unsuccessful authentication attempts for each authentication mechanism corresponds to the configured maximum and that the critical authentication mechanisms cause the device to wipe. Alternatively, if the number of authentication failures is not maintained for the interface under test, the evaluator shall verify that upon booting the TOE between unsuccessful authentication attempts another authentication factor interface is presented before the interface under test.

Summary

The evaluator attempted to unlock the device with the wrong password three times (the maximum number of tries allowed as configured for the test) and confirmed the device was then wiped and all data made inaccessible. In a subsequent test, the evaluator power cycled the device in between the second and third attempt and confirmed after the reboot and the third unlock attempt, the device was wiped.

2.1.4.2 Extended: Bluetooth User Authorization (FIA_BLT_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FIA_BLT_EXT.1-ASE-01

The evaluator shall examine the TSS to ensure that it contains a description of when user permission is required for Bluetooth pairing, and that this description mandates explicit user authorization via manual input for all Bluetooth pairing, including application use of the Bluetooth trusted channel and situations where temporary (non-bonded) connections are formed.

Summary

Section 8.8.3 in the [ST] [\[1\]](#) describes the *Bluetooth* protocol.

The evaluator verified that section 8.8.3 specifically addresses how a user authorizes Bluetooth pairing through the Bluetooth interface in the settings of iOS. This section also has the following statement: During pairing time, another will send a request to pair, and commonly, 6 digits are displayed on both sides (the iOS device and the device which wishes to be paired). The user has to manually match these digits and then accept the pairing. Whenever a device does not offer an automatic exchange of a PIN, a window is shown and the user is asked to enter a PIN.

Guidance Assurance Activities

Assurance Activity AA-FIA_BLT_EXT.1-AGD-01

The evaluator shall examine the API documentation provided according to Section 5.2.2 and verify that this API documentation does not include any API for programmatic entering of pairing information (e.g. PINs, numeric codes, or "yes/no" responses) intended to bypass manual user input during pairing.

Summary

The evaluator examined the provided API documentation and determined that it does not include any API for programmatic entering of pairing information intended to bypass manual user input during pairing. Also, per [CCGUIDE] [\[1\]](#) section 3.4.4 *Bluetooth Configuration*, manual authorization is implicitly configured since pairing can only occur when explicitly authorized through the Settings>>Bluetooth interface.

Assurance Activity AA-FIA_BLT_EXT.1-AGD-02

The evaluator shall examine the AGD guidance to verify that these user authorization screens are clearly identified and instructions are given for authorizing Bluetooth pairings.

Summary

[CCGUIDE] [\[1\]](#) section 3.4.4 *Bluetooth Configuration* provides information for configuring Bluetooth which references the iPhone and iPad User Guides [iPhone_UG] [\[1\]](#) and [iPad_UG] [\[1\]](#) section *Connect bluetooth devices* for how to turn Bluetooth on and off and how to pair and un-pair a Bluetooth device. This involves turning on the Bluetooth feature on the TOE device (go to Settings>>Bluetooth), choose the Bluetooth accessory, and then enter a passkey or PIN if prompted. When pairing is complete, user can use the Bluetooth accessory with their iOS device. Also, Bluetooth can be disassociated via the Control Center.

Additionally, [CCGUIDE] [\[1\]](#) section 3.4.4 states the following:

"

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.

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).

"

Test Activities

Assurance Activity AA-FIA_BLT_EXT.1-ATE-01

The evaluator shall perform the following test:

- **Test 1:** *The evaluator shall perform the following steps:*
 - *Step 1: Initiate pairing with the TOE from a remote Bluetooth device that requests no man-in-the-middle protection, no bonding, and claims to have NoInputNoOutput input-output (IO) capability. (Such a device will attempt to evoke behavior from the TOE that represents the minimal level of user interaction that the TOE supports during pairing.)*
 - *Step 2: Verify that the TOE does not permit any Bluetooth pairing without explicit authorization from the user (e.g. the user must have to minimally answer “yes” or “allow” in a prompt).*

Summary

The evaluator initiated a Bluetooth pairing as described and verified that a PIN is shown to the user and approval is required before pairing is achieved.

2.1.4.3 Extended: Bluetooth Mutual Authentication (FIA_BLT_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FIA_BLT_EXT.2.1-ASE-01

The evaluator shall ensure that the TSS describes how data transfer of any type is prevented before the Bluetooth pairing is completed. The TSS shall specifically call out any supported RFCOMM and L2CAP data transfer mechanisms. The evaluator shall ensure that the data transfers are only completed after the Bluetooth devices are paired and mutually authenticated.

Summary

Section 8.8.3 of the [ST]  describes the *Bluetooth* protocol.

The evaluator ensured that section 8.8.3 describes that the Bluetooth pairing how to be complete before data can be exchanged. The evaluator also verified that the TSS do not call out any RFCOMM or L2CAP data transfer mechanism. Paragraph 2 of section 8.8.3 describes how the device have to authenticate for the pairing to successfully complete, and data to be exchanged.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FIA_BLT_EXT.2.1-ATE-01

The evaluator shall perform the following test:

- **Test 1:** *The evaluator shall use a Bluetooth tool to attempt to access TOE files using the OBEX Object Push service and verify that pairing and mutual authentication are required by the TOE before allowing access. (If the OBEX Object Push service is unsupported on the TOE, a different service that transfers data over Bluetooth L2CAP and/or RFCOMM may be used in this test.)*

Summary

Test performed for FIA_BLT_EXT.1 demonstrates this functionality as well: data can only be communicated once a connection is established. Such connection can only be established when devices are either manually or automatically paired. Automatic pairing can only be enabled on the TOE after an initial manual pairing was successful. The evaluator opened the Bluetooth configuration to make the device visible while a previously connected device was visible to the TOE and verified that the connection was established and the device was in a "Connected" state.

2.1.4.4 Rejection of Duplicate Bluetooth Connections (FIA_BLT_EXT.3)

TSS Assurance Activities

Assurance Activity AA-FIA_BLT_EXT.3.1-ASE-01

The evaluator shall ensure that the TSS describes how Bluetooth connections are maintained such that two devices with the same Bluetooth device address are not simultaneously connected and such that the initial connection is not superseded by any following connection attempts.

Summary

Section 8.8.3 in the [ST] describes the *Bluetooth* protocol within iOS.

Section 8.8.3 describes that a local database is kept of all Bluetooth device addresses for paired devices which is checked prior to any automatic connection attempt. Bluetooth cannot establish more than one connection. Multiple connection attempts from the same BD_ADDR for an established connection will be discarded.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FIA_BLT_EXT.3.1-ATE-01

The evaluator shall perform the following test:

- **Test 1:** *The evaluator shall perform the following steps:*
 - *Step 1: Make a Bluetooth connection between the TOE and a remote Bluetooth device with address a known address (BD_ADDR1).*
 - *Step 2: Attempt a connection to the same TOE from a second remote Bluetooth device claiming to have a Bluetooth device address matching BD_ADDR1.*
 - *Step 3: Using a Bluetooth protocol analyzer, verify that the second connection attempt is ignored by the TOE and that the initial connection to the device with BR_ADDR1 is unaffected.*

Summary

The evaluator paired the TOE with another device and got the Bluetooth ID of that device. He then used that information to configure another device with the same Bluetooth ID and attempt to impersonate the connection. He verified that the connection failed and collected Wireshark data showing the reason was "Connection Already Exists."

2.1.4.5 Extended: Secure Simple Pairing (FIA_BLT_EXT.4)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FIA_BLT_EXT.4-ATE-01

- **Test 1:** *The evaluator shall perform the following steps:*
 - *Step 1: Initiate pairing with the TOE from a remote Bluetooth device that supports Secure Simple Pairing.*
 - *Step 2: During the pairing process, observe the packets in a Bluetooth protocol analyzer and verify that the TOE claims support for both "Secure Simple Pairing (Host Support)" and "Secure Simple Pairing (Controller Support)" during the LMP Features Exchange.*
 - *Step 3: Verify that Secure Simple Pairing is used during the pairing process.*

Summary

The evaluator used a Bluetooth sniffer to collect traffic during a Bluetooth pairing operation and verified that Secure Simple Pairing is used during the pairing process.

2.1.4.6 Extended: Password Management (FIA_PMG_EXT.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-FIA_PMG_EXT.1-AGD-01

The evaluator shall examine the operational guidance to determine that it provides guidance to security administrators on the composition of strong passwords, and that it provides instructions on setting the minimum password length.

Summary

[CCGUIDE] [\[1\]](#) section 3.4.1 *Passcode Authentication Configuration* provides guidance to administrators on the composition of strong passcode which consists of:

- Passcode composing of any combination of upper and lower case letters, numbers, and special characters: "!", "@", "#", "\$", "%", "^", "&", "*", "(", ")";
- Passcode length up to 16 characters

Passcode policy is defined by administrator using the Passcode Policy Payload in a Configuration Profile described in [IOS_CFG] [\[1\]](#). 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 RequireAlphanumeric 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.

The Passcode Policy Payload is outlined in Table 7 in [CCGUIDE] [\[1\]](#).

Test Activities

Assurance Activity AA-FIA_PMG_EXT.1-ATE-01

The evaluator shall also perform the following tests. Note that one or more of these tests can be performed with a single test case.

- **Test 1:** The evaluator shall compose passwords that either meet the requirements, or fail to meet the requirements, in some way. For each password, the evaluator shall verify that the TOE supports the password. While the evaluator is not required (nor is it feasible) to test all possible compositions of passwords, the evaluator shall ensure that all characters, rule characteristics, and a minimum length listed in the requirement are supported, and justify the subset of those characters chosen for testing.

Summary

The evaluator configured password quality rules using the Apple Configurator 2 and proceeded to use passwords that both did and did not conform to those rules. All acceptances and failures were as expected.

2.1.4.7 Extended: Authentication Throttling (FIA_TRT_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FIA_TRT_EXT.1-ASE-01

The evaluator shall verify that the TSS describes the method by which authentication attempts are not able to be automated. The evaluator shall ensure that the TSS describes either how the TSF disables authentication via external interfaces (other than the ordinary user interface) or how authentication attempts are delayed in order to slow automated entry and shall ensure that this delay totals at least 500 milliseconds over 10 attempts for all authentication mechanisms selected in FIA_UAU.5.1.

Summary

Section 8.4 in the [ST] [1](#) describes the *Identification and Authentication (FIA)*.

The evaluator reviewed this section and determined that whenever a passcode is entered, iOS forces the PBKDF2 function to iterate for 80 milliseconds. In addition, the TSF enforce a delay of 5 seconds between repeated failed authentication.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.4.8 Multiple Authentication Mechanisms (FIA_UAU.5)

TSS Assurance Activities

Assurance Activity AA-FIA_UAU.5-ASE-01

The evaluator shall ensure that the TSS describes each mechanism provided to support user authentication and the rules describing how the authentication mechanism(s) provide authentication.

Summary

Section 8.4 in the [ST] [1](#) describes the *Identification and Authentication (FIA)*.

The evaluator reviewed this section and verified that the passcode, Face ID and Touch ID are the only way that the user can authenticate, as described in section 8.4 and as claimed in FIA_UAU.5. They can however not be used as two-factor authentication.

Guidance Assurance Activities

Assurance Activity AA-FIA_UAU.5-AGD-01

The evaluator shall verify that configuration guidance for each authentication mechanism is addressed in the AGD guidance.

Summary

Guidance for authentication is provided in section 3.4 *Identification & Authentication Configuration* of [CCGUIDE] covering configuration for authentication via passcode, biometrics, Bluetooth, and X.509 certificate. Additional information is provided to configure failed authentication attempts, re-authentication as well as certificate validation and device enrollment.

The evaluator determined the provided guidance contains sufficient detail for each authentication mechanism. Section 3.4.1 *Passcode Authentication Configuration* describes how to configure the passcode authentication which uses the Passcode Policy Payload described in this section and further in [IOS_CFG]. Section 3.4.2 *Biometric Authentication Factors* describes how to configure Touch ID and Face ID using the instructions provided in [iPhone_UG] and [iPad_UG]. Section 3.4.4 *Bluetooth Configuration* describes how to pair the TOE and another Bluetooth device using the instructions provided in [iPhone_UG] and [iPad_UG]. Section 3.4.7 *X.509 Certificate Configuration* describes how to configure X.509 authentication a Configuration Profile described in section "Certificate Payload" of [IOS_CFG].

The evaluator thus determined that the guidance provides instructions to configure all of the authentication mechanisms.

Test Activities

Assurance Activity AA-FIA_UAU.5-ATE-01

- **Test 1:** For each authentication mechanism selected, the evaluator shall enable that mechanism and verify that it can be used to authenticate the user at the specified authentication factor interfaces.
- **Test 2:** For each authentication mechanism rule, the evaluator shall ensure that the authentication mechanism(s) behave accordingly.

Summary

Testing of password authentication was performed for FIA_AFL_EXT.1.

The evaluator verified both biometric authentication mechanisms, Touch ID and Face ID, by adding the appropriate authentication for the device, locking the screen and verifying that the device unlocks using the biometric mechanism.

2.1.4.9 Extended: Accuracy of Biometric Authentication (FIA_BMG_EXT.1)

FIA_BMG_EXT.1.1

TSS Assurance Activities

Assurance Activity AA-FIA_BMG_EXT.1.1-ASE-01

The evaluator shall verify that the TSS contains evidence supporting the testing and calculations completed to determine the FAR and FRR.

Appendix H provides guidance to how this testing could be completed and to what error bars are expected when the Rule of 3 is applied. The evaluator shall consult Appendix H as a reference, but should not treat it as a mandate. The evaluator shall verify that the TSS contains evidence of how many imposters were used for testing, whether online or offline testing was used and if offline testing was completed, evidence describing the differences between the biometric system used for testing and the TOE in the evaluated configuration, if any. The evaluator shall verify that the testing describes how imposters are compared to enrolled users, for example, if multiple devices for online testing or full cross-comparison for offline testing was used. Adequate documentation is required to demonstrate that testing was completed to support the claimed FAR and FRR.

Summary

The evaluator reviewed section 8.4.1 which contains the *Biometric Authentication* information from the TOE. Specifically, section 8.4.1.1 and 8.4.1.3 explain the accuracy of biometric authentication for the TOE. The evaluator could find in these sections evidence of the calculation used to determine the FAR and FRR for the TOE. The evaluator verified that: the testing was performed offline; that the system used for testing is emulated on a different platform in a cloud computation infrastructure for efficiency reasons; a full cross-comparison for offline testing is used to compare imposters for gen.1 sensors, and a partial full-cross comparison is used for gen.2 and gen.3 sensors.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

FIA_BMG_EXT.1.2

TSS Assurance Activities

Assurance Activity AA-FIA_BMG_EXT.1.2-ASE-01

The evaluator shall verify that the TSS indicates which SAFAR the TOE is targeting and contains evidence supporting the calculations, per Appendix H.3, completed to determine the SAFAR. The evaluator shall verify that the TSS contains evidence of how the authentication factors interact, per FIA_UAU.5.2 and FIA_AFL_EXT.1. The evaluator shall verify that the TSS, contains the combination(s) of authentication factors needed to meet the SAFAR, and the number of attempts for each authentication factor the TOE is configured to allow. Adequate documentation is required to demonstrate the calculations completed to support the claimed SAFAR.

Summary

The evaluator reviewed section 8.4.1 of the [ST][\[ST\]](#) which describes then biometric authentication supported by the TOE. The TOE supports fingerprint (Touch ID) or facial recognition authentication (Face ID). Sections 8.4.1.1 and 8.4.1.3 of the ST describes the *Accuracy of Biometric Authentication*.

Section 8.4 describes how the Face ID and Touch ID interact with the passcode set on the device. A device that supports Face ID does not support Touch ID and vice-versa. A passcode must be supplied in certain cases for additional security.

- 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 156 hours (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.
- After initiating power off/Emergency SOS.

Section 8.4.1.3 describes that there can be at most only 5 attempts for the fingerprint and 5 attempts for the facial recognition. Passed these attempts, at most 10 attempts for the 6-digit passcode are provided.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.4.10 Extended: Biometric Enrollment (FIA_BMG_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FIA_BMG_EXT.2-ASE-01

The evaluator shall verify that the TSS describes how the quality of samples used to create the authentication template at enrollment are verified. As well as the quality standard that the validation method uses to perform the assessment.

Summary

Section 8.4.1.4 of the TSS describes the biometric sample quality used on the TOE for deriving the authentication template enrolled. This includes:

- deciding what portion of the sensor is covered by the finger. Sensor regions containing very weak signal are considered not covered. Samples with high number of such regions are rejected
- assessing the level of residual fixed-pattern noise. Samples where the noise could significantly alter the fingerprint pattern are rejected
- detection and removal of regions affected by image discontinuity caused by finger motion. Samples with many regions affected by motion are rejected

Moreover, the testing mechanisms are repeated for every major iOS release. Section 8.4.1.1 describe the tests subjects chosen for the samples: various people were used to collect facial expressions with various eye wear, indoor/outdoor settings, age, ethnicity and gender. Family members were also chosen, such as siblings.

The method used to perform this assessment is an off-line test. Fingerprint data (Touch ID) were collected separately for each sensor. The data was passed to an emulator for efficiency reasons, and not on the production hardware. Note that a special testing step is performed in order to guarantee that the results on the emulator match the results on the production hardware. For gen.1 sensors, a full-cross comparison scheme is used. For gen.2 and gen.3 sensors, a partial cross-comparison is used. For Face ID, a similar methodology is used using offline testing on macOS systems. A full-cross comparison was used of all subjects in the P3 and P4 datasets.

Guidance Assurance Activities

Assurance Activity AA-FIA_BMG_EXT.2-AGD-01

The evaluator shall verify that the AGD guidance describes how to enroll a user for each biometric modality supported.

Summary

[CCGUIDE] section 3.4.2 *Biometric Authentication Factors* describes how to enroll biometric authentication factors which explicitly references [iPhone_UG] and [iPad_UG]. It states the following instructions from these user guides:

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.

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 and tapping the "Reset Face ID" option.

Test Activities

Assurance Activity AA-FIA_BMG_EXT.2-ATE-01

The evaluator shall input biometric samples for enrollment. Upon inputting biometric samples a fixed number of times as specified in the prompts, one or more authentication templates will be generated. The evaluator shall verify that the device only accepts samples of sufficient quality or requests additional samples if the authentication template is not of sufficient quality. For all quality metrics, the evaluator shall ensure that biometric samples achieving a worse quality score than the prescribed threshold are rejected.

Summary

The evaluator enrolled in both Touch ID and Face ID on the appropriate TOE device following the instructions on the device. For each biometric method, the evaluator made attempts to unlock the device while disrupting the input and found access was not granted.

2.1.4.11 Extended: Biometric Verification (FIA_BMG_EXT.3)

TSS Assurance Activities

Assurance Activity AA-FIA_BMG_EXT.3-ASE-01

The evaluator shall verify that the TSS describes how the quality of samples used to verify authentication are verified. As well as the quality standard that the validation method uses to perform the assessment.

Summary

Section 8.4.1.4 of the TSS describes the biometric sample quality used on the TOE for deriving the authentication template enrolled. This includes:

For Touch ID:

- deciding what portion of the sensor is covered by the finger. Sensor regions containing very weak signal are considered not covered. Samples with high number of such regions are rejected

- assessing the level of residual fixed-pattern noise. Samples where the noise could significantly alter the fingerprint pattern are rejected
- detection and removal of regions affected by image discontinuity caused by finger motion. Samples with many regions affected by motion are rejected

For Face ID:

- User is attending the device
- No significant depth holes in the depth map
- Anti-Spoofing network to reject physical spoofs
- For enrollment
 - No occlusions detected (e.g. hand covering face)
 - User attending device
 - No significant depth holes in the depth map
 - Anti-spoofing network to reject physical spoofs

Moreover, the TSS describe that:

"The test is based specialized datasets containing different levels of coverage and different artifacts. These samples are fed to the biometric system and it is confirmed whether the sample is correctly passed or rejected from the processing as expected. Additionally, the biometric system is tested by feeding artificially created images containing different geometric patterns."

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FIA_BMG_EXT.3-ATE-01

The evaluator shall enroll a user for each biometric modality supported. The evaluator will then input biometric samples for verification and ensure that the device only accepts samples of sufficient quality. The evaluator shall ensure that biometric samples achieving a worse quality score than the prescribed threshold are rejected.

Summary

The evaluator enrolled in both Touch ID and Face ID providing several instances of disrupted input and found that the poor data samples were not accepted by the TOE for use as valid biometric data.

2.1.4.12 Extended: Handling Unusual Biometric Templates (FIA_BMG_EXT.5)

TSS Assurance Activities

Assurance Activity AA-FIA_BMG_EXT.5-ASE-01

The evaluator shall verify that the TSS how the matching algorithm addresses properly formatted templates with unusual data properties, incorrect syntax, or low quality.

Summary

See work unit AA-FIA_BMG_EXT.3-ASE-01.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FIA_BMG_EXT.5-ATE-01

[The evaluator shall verify that the TSS how the matching algorithm addresses properly formatted templates with unusual data properties, incorrect syntax, or low quality.] The evaluator shall ensure that these claims are sound through appropriate testing based on test programs provided by the vendor.

Summary

The evaluator found the claims specified in section 8.4.1 of [ST] to be sound based on his experience testing the biometric features of the TOE.

2.1.4.13 Re-Authentication (FIA_UAU.6)

FIA_UAU.6.1

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FIA_UAU.6.1-1-ATE-01

- **Test 1:** The evaluator shall configure the TSF to use the Password Authentication Factor according to the AGD guidance. The evaluator shall change Password Authentication Factor according to the AGD guidance and verify that the TSF requires the entry of the Password Authentication Factor before allowing the factor to be changed.
- **Test 2:**[conditional] For each BAF selected in FIA_UAU.5.1, the evaluator shall configure the TSF to use the BAF, which includes configuring the Password Authentication Factor, according to the AGD guidance. The evaluator shall change the BAF according to the AGD guidance and verify that the TSF requires the entry of the Password Authentication Factor before allowing the BAF to be changed.
- **Test 3:**[conditional] If “hybrid” is selected in FIA_UAU.5.1, the evaluator shall configure the TSF to use the BAF and PIN or password, which includes configuring the Password Authentication Factor, according to the AGD guidance. The evaluator shall change the BAF and PIN according to the AGD guidance and verify that the TSF requires the entry of the Password Authentication Factor before allowing the factor to be changed.

Summary

The evaluator verified that the current passcode is required to change any authentication factor: passcode, Touch ID, or Face ID.

FIA_UAU.6.2

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FIA_UAU.6.1-2-ATE-01

- **Test 1:** The evaluator shall configure the TSF to transition to the locked state after a time of inactivity (FMT_SMF_EXT.1) according to the AGD guidance. The evaluator shall wait until the TSF locks and then verify that the TSF requires the entry of the Password Authentication Factor before transitioning to the unlocked state.
- **Test 2:** [conditional] For each BAF selected in FIA_UAU.5.1, the evaluator shall repeat Test 1 verifying that the TSF requires the entry of the BAF before transitioning to the unlocked state.
- **Test 3:** [conditional] If "hybrid" is selected in FIA_UAU.5.1, the evaluator shall repeat Test 1 verifying that the TSF requires the entry of the BAF and PIN/password before transitioning to the unlocked state.
- **Test 4:** The evaluator shall configure user-initiated locking according to the AGD guidance. The evaluator shall lock the TSF and then verify that the TSF requires the entry of the Password Authentication Factor before transitioning to the unlocked state.
- **Test 5:** [conditional] For each BAF selected in FIA_UAU.5.1, the evaluator shall repeat Test 4 verifying that the TSF requires the entry of the BAF before transitioning to the unlocked state.
- **Test 6:** [conditional] If "hybrid" is selected in FIA_UAU.5.1, the evaluator shall repeat Test 4 verifying that the TSF requires the entry of the BAF and PIN/password before transitioning to the unlocked state.

Summary

The evaluator confirmed that an authentication factor (passcode, Touch ID, or Face ID) is required to unlock the device after both inactivity lock and user-initiated lock.

2.1.4.14 Protected Authentication Feedback (FIA_UAU.7)

TSS Assurance Activities

Assurance Activity AA-FIA_UAU.7-ASE-01

The evaluator shall ensure that the TSS describes the means of obscuring the authentication entry, for all authentication methods specified in FIA_UAU.5.1.

Summary

Section 8.4 in the [ST] [describes the Identification and Authentication \(FIA\)](#).

The TSS describes that each character of the passcode, when entered, replaced by an dot symbol. Moreover, section 8.4 states that

"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."

Guidance Assurance Activities

Assurance Activity AA-FIA_UAU.7-AGD-01

The evaluator shall verify that any configuration of this requirement is addressed in the AGD guidance and that the password is obscured by default.

Summary

[CCGUIDE] section 3.4.5 *Protected Authentication Feedback Configuration* provides information on protected authentication feedback. It states the following:

In the evaluated configuration, the passcode on the TOE devices is obscured by default. No configuration is needed.

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. If three invalid biometric samples are presented the device will offer passcode entry. After five invalid biometric samples are presented passcode authentication is required.

For Face ID, when an invalid facial sample is given or cannot be authenticated, the user needs to swipe up before a second attempt can occur and passcode entry will be presented to the user as an option. After five invalid Face ID attempts, the device will vibrate and passcode entry must be used.

Test Activities

Assurance Activity AA-FIA_UAU.7-ATE-01

- **Test 1:** The evaluator shall enter passwords on the device, including at least the Password Authentication Factor at lockscreen, and verify that the password is not displayed on the device.
- **Test 2:**[conditional] For each BAF selected in FIA_UAU.5.1, the evaluator shall authenticate by producing a biometric sample at lockscreen. As the biometric algorithms are performed, the evaluator shall verify that sensitive images, audio, or other information identifying the user are kept secret and are not revealed to the user. Additionally, the evaluator shall produce a biometric sample that fails to authenticate and verify that the reason(s) for authentication failure (user mismatch, low sample quality, etc.) are not revealed to the user. It is acceptable for the BAF to state that it was unable to physically read the biometric sample, for example, if the sensor is unclear or the biometric sample was removed too quickly. However, specifics regarding why the presented biometric sample failed authentication shall not be revealed to the user.

Summary

The evaluator entered the password in the lock screen window and verified that obscured feedback was provided. For a fraction of a second, the key pressed is displayed to enable the user to verify the intended key was touched. The evaluator accepts that this does not violate the requirement.

The evaluator used the biometric fingerprint authentication mechanism and found no sensitive information to be displayed. Also, no reason for failure was provided when attempting to use biometric fingerprint authentication using an unrecognized fingerprint.

The evaluator used the face authentication mechanism and found no sensitive information to be displayed. Also, no reason for failure was provided when attempting to use biometric face authentication using an unrecognized face.

2.1.4.15 Extended: Authentication for Cryptographic Operation (FIA_UAU_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FIA_UAU_EXT.1-ASE-01

The evaluator shall verify that the TSS section of the ST describes the process for decrypting protected data and keys. The evaluator shall ensure that this process requires the user to enter a Password Authentication Factor and, in accordance with FCS_CKM_EXT.3, derives a KEK, which is used to protect the software-based secure key storage and (optionally) DEK(s) for sensitive data, in accordance with FCS_STG_EXT.2.

Summary

Section 8.2.1 in the [ST] describes the *Overview of Key Management*.

Section 8.2.1 describes that the passcode is used, combined with the UID and the salt, to derive the passcode key (KEK) which is then used to decrypt the class keys for protected data. That passcode key is erased whenever the device is locked, and reconstructed whenever the device is unlocked with the right passcode.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FIA_UAU_EXT.1-ATE-01

The following tests may be performed in conjunction with FDP_DAR_EXT.1 and FDP_DAR_EXT.2.

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

- **Test 1:** *The evaluator shall enable encryption of protected data and require user authentication according to the AGD guidance. The evaluator shall write, or the developer shall provide access to, an application that includes a unique string treated as protected data. The evaluator shall reboot the device, use a tool provided by developer to search for the unique string amongst the application data, and verify that the unique string cannot be found. The evaluator shall enter the Password Authentication Factor to access full device functionality, use a tool provided by the developer to access the unique string amongst the application data, and verify that the unique string can be found.*
- **Test 2:** *[conditional] The evaluator shall require user authentication according to the AGD guidance. The evaluator shall store a key in the software-based secure key storage. The evaluator shall lock the device, use a tool provided by developer to access the key amongst the stored data, and verify that the key cannot be retrieved or accessed. The evaluator shall enter the Password Authentication Factor to access full device functionality, use a tool provided by developer to access the key, and verify that the key can be retrieved or accessed.*
- **Test 3:** *[conditional] The evaluator shall enable encryption of sensitive data and require user authentication according to the AGD guidance. The evaluator shall write, or the developer shall provide access to, an application that includes a unique string treated as sensitive data. The evaluator shall lock the device, use a tool provided by developer to attempt to access the unique string amongst the application data, and verify that the unique string cannot be found. The evaluator shall enter the Password Authentication Factor to access full device functionality, use a tool provided by developer to access the unique string amongst the application data, and verify that the unique string can be retrieved.*

Summary

The described tests cannot be performed on Apple devices due to the method of data encryption implemented as described in FDP_DAR_EXT.1. For all three of these tests, the debug session showing the life cycle of the class keys discussed in FCS_CKM_EXT.4 satisfies this test requirement by showing that the keys needed to decrypt data are not available prior to completing the defined key unlock steps.

This approach has been accepted in previous iOS evaluations.

2.1.4.16 Extended: Timing of Authentication (FIA_UAU_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FIA_UAU_EXT.2-ASE-01

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

Summary

Section 8.4 of the [ST]  describes the *Identification and Authentication (FIA)*.

The first paragraph of section 8.4 explains that except for making emergency calls, using the camera and the flashlight, users need to authenticate to the device to perform any other services.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FIA_UAU_EXT.2-ATE-01

The evaluator shall attempt to perform some actions not listed in the selection while the device is in the locked state and verify that those actions do not succeed.

Summary

The evaluator waited until the device locked automatically and verified that no actions other than ones listed in the selection could be performed.

2.1.4.17 Extended: Validation of Certificates (FIA_X509_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FIA_X509_EXT.1-ASE-01

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

Summary

Section 8.4.2 [ST]  describes *Certificates*.

The evaluator reviewed section 8.4.2 and verified that the verification of certificates takes place whenever the TOE tries to establish a connection using a remote certificate. The process for certificate validation is the following:

"When attempting to establish a connection using a remote certificate, the certificate is first checked to ensure it is valid. Certificates are validated against the common name (CN) portion of the distinguished name (DN). If the CN does not match the corresponding domain name system (DNS) or IP Address of the server being accessed, validation and subsequently the connection will fail. If the certificate is valid, the attempt to establish the connection continues. If the certificate is invalid, the next step is up to the application. The application should provide an indication to the user that the certificate is invalid and options to accept or reject."

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FIA_X509_EXT.1-ATE-01

The tests described must be performed in conjunction with the other Certificate Services assurance activities, including the use cases in FIA_X509_EXT.2.1 and FIA_X509_EXT.3. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. The evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA.

- **Test 1:** The evaluator shall then load a certificate or certificates to the Trust Anchor Database needed to validate the certificate to be used in the function (e.g. application validation, trusted channel setup, or trusted software update), and demonstrate that the function succeeds. The evaluator then shall delete one of the certificates, and show that the function fails.
- **Test 2:** The evaluator shall demonstrate that validating an expired certificate results in the function failing.
- **Test 3:** The evaluator shall test that the TOE can properly handle revoked certificates-conditional on whether CRL or OCSP is selected; if both are selected, then a test shall be performed for each method. The evaluator shall test revocation of the node certificate and revocation of the intermediate CA certificate (i.e. the intermediate CA certificate should be revoked by the root CA). For the test of the WLAN use case, only pre-stored CRLs are used. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails.
- **Test 4:** The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOE's certificate does not contain the basicConstraints extension. The validation of the certificate path fails.
- **Test 5:** The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOE's certificate has the cA flag in the basicConstraints extension not set. The validation of the certificate path fails.
- **Test 6:** The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOE's certificate has the cA flag in the basicConstraints extension set to TRUE. The validation of the certificate path succeeds.
- **Test 7:** The evaluator shall modify any byte in the first eight bytes of the certificate and demonstrate that the certificate fails to validate (the certificate will fail to parse correctly).
- **Test 8:** The evaluator shall modify any bit in the last byte of the signature algorithm of the certificate and demonstrate that the certificate fails to validate (the signature on the certificate will not validate).
- **Test 9:** The evaluator shall modify any byte in the public key of the certificate and demonstrate that the certificate fails to validate (the signature on the certificate will not validate).

Summary

Test 1 was performed for FCS_TLSC_EXT.1.1 and FCS_TLSC_EXT.1.2. These tests demonstrated that certificates signed by a CA known to the TOE are trusted by the TOE and valid certificates signed by a CA unknown to the TOE are treated as untrusted by the TOE.

The remaining 8 tests performed show various types of certificate failures, including checking for expired or revoked certificates, certificates with incorrectly defined extensions, certificates signed by an untrusted CA, and malformed certificates.

2.1.4.18 Extended: X509 Certificate Authentication (FIA_X509_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FIA_X509_EXT.2-ASE-01

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

Summary

Section 8.4.2 [ST][\[1\]](#) describes the *Certificates*.

Section 8.4.2 describes that every certificate in iOS is assigned a certificate type respective to the area in which this certificate is going to be used (such as AppleX509Basic or AppleSSL for example), as described in [CCGUIDE][\[1\]](#).

Assurance Activity AA-FIA_X509_EXT.2-ASE-02

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

Summary

Section 8.4.2 [ST][\[1\]](#) describes the *Certificates*.

Whenever a certificate cannot be validated, it is up to the application requesting the connection to be established to decide whether to abort the connection or not.

Guidance Assurance Activities

Assurance Activity AA-FIA_X509_EXT.2-AGD-01

If the requirement that the administrator is able to specify the default action, then the evaluator shall ensure that the operational guidance contains instructions on how this configuration action is performed.

Summary

The default setting (which is for the TOE to not accept untrusted certificates) according to [ST][\[1\]](#) section 8.8.1 *EAP-TLS and TLS*. [CCGUIDE][\[1\]](#) section 3.4.7.1 *Certificate Validation* states that 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 certificate is presented.

Test Activities

Assurance Activity AA-FIA_X509_EXT.2-ATE-01

The evaluator shall perform the following test for each trusted channel:

- **Test 1:** *The evaluator shall demonstrate that using a valid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate, and observe that the action selected in FIA_X509_EXT.2.2 is performed. If the selected action is administrator-configurable, then the evaluator shall follow the operational guidance to determine that all supported administrator-configurable options behave in their documented manner.*

Summary

The evaluator performed tests showing successful CA chain validation, unsuccessful CA chain validation with user selectable acceptance of trust, and unsuccessful CA chain validation with administrator defined rejection of trust.

2.1.4.19 Extended: Request Validation of certificates (FIA_X509_EXT.3)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-FIA_X509_EXT.3-AGD-01

The evaluator shall verify that the API documentation provided according to Section 5.2.2 includes the security function (certificate validation) described in this requirement. This documentation shall be clear as to which results indicate success and failure.

Summary

[CCGUIDE] [\[1\]](#) section 3.4.7.1 *Certificate Validation* references "Certificate, Key, and Trust Services Reference" [CKTSREF] [\[1\]](#) as the API documentation related to certificate validation. This is covered by the function 'SetTrustEvaluate'.

Test Activities

Assurance Activity AA-FIA_X509_EXT.3-ATE-01

The evaluator shall write, or the developer shall provide access to, an application that requests certificate validation by the TSF. The evaluator shall verify that the results from the validation match the expected results according to the API documentation. This application may be used to verify that import, removal, modification, and validation are performed correctly according to the tests required by FDP_STG_EXT.1, FTP_ITC_EXT.1, FMT_SMF_EXT.1.1, and FIA_X509_EXT.1.

Summary

The evaluator configured a TLS server to accept only one cipher at a time. For each cipher, he used a web browser on the TOE to connect to the server. For each cipher suite, the TOE was able to establish a connection.

2.1.4.20 Extended: Enrollment of Mobile Device into Management (FIA_ENR_EXT.2(AGENT))

TSS Assurance Activities

Assurance Activity AA-FIA_ENR_EXT.2-MDMA-ASE-01

The evaluator shall examine the TSS to verify that it describes which types of reference identifiers are acceptable and how the identifier is specified (e.g. preconfigured in the MDM Agent, by the user, by the MDM server, in a policy).

Summary

Section 8.4.3 of the [ST] [\[1\]](#) specifies the *MDM Server Reference ID*.

Table 12 in section 8.4.3 specifies the reference identifiers used by the MDM server. They can be the following:

- `serve_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
- `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

The evaluator verified which type of reference identifier are acceptable to the TOE, and they configured by the MDM enrollment service through JSON tags on the mobile device.

Guidance Assurance Activities

Assurance Activity AA-FIA_ENR_EXT.2-MDMA-AGD-01

The evaluator shall examine the operational guidance to verify that it describes how to configure reference identifier of the MDM Server's certificate and, if different than the reference identifier, the Domain Name or IP address (for connectivity) of the MDM Server.

Summary

[CCGUIDE][\[1\]](#) section 3.2.12 *Configure MDM Agent and MDM Communications* describes how to configure MDM Agent and MDM Server communications as follows.

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.

Test Activities

Assurance Activity AA-FIA_ENR_EXT.2-MDMA-ATE-01

The evaluator shall follow the operational guidance to establish the reference identifier of the MDM server on the MDM Agent and in conjunction with other Assurance Activities verify that the MDM Agent can connect to the MDM Server and validate the MDM Server's certificate.

Summary

The evaluator verified the Device Identity Certificate contains the issuer certificate and that the Server URL and Check-in URL point to the established server host.

2.1.4.21 Port Access Entity Authentication (FIA_PAE_EXT.1(WLAN))

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FIA_PAE_EXT.1-WLAN-ATE-01

The evaluator shall perform the following tests:

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

Summary

The evaluator performed all 3 tests as described and found that in test 1, the TOE connected as expected and in tests 2 and 3, the connection failed as expected.

2.1.4.22 X.509 Certificate Authentication (EAP-TLS) (FIA_X509_EXT.2(WLAN))


TSS Assurance Activities

Assurance Activity AA-FIA_X509_EXT.2-WLAN-ASE-01

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

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

Summary

Section 8.8.1 *EAP-TLS and TLS* in the [ST]  describes the EAP-TLS and TLS protocols and ciphersuites supported by the TOE.

Section 8.8.1 describes that when the TOE is configured to used EAP-TLS, 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. When the TLSAllowTrustExceptions key in the

Wi-Fi payload is used, the administrator can enforce that untrusted certificates are not accepted and the authentication fails if such an untrusted certificate is presented. Please refer to [CCGUIDE] for a description of the default action that the administrator is able to specify.

Guidance Assurance Activities

Assurance Activity AA-FIA_X509_EXT.2-WLAN-AGD-01

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

Summary

[CCGUIDE] section 3.4.7 X.509 Certificate Configuration states the following:

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:

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

Also, 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.

Test Activities

Assurance Activity AA-FIA_X509_EXT.2-WLAN-ATE-01

The evaluator shall perform the following test for each trusted channel:

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

Summary

The evaluator performed numerous test demonstrating successful CA chain validation, unsuccessful CA chain validation with user selectable acceptance of trust, and unsuccessful CA chain validation with administrator defined rejection of trust.

2.1.5 Security management (FMT)

2.1.5.1 Extended: Management of Security Functions Behavior (FMT_MOF_EXT.1)

FMT_MOF_EXT.1.1

TSS Assurance Activities

Assurance Activity AA-FMT_MOF_EXT.1.1-ASE-01

The evaluator shall verify that the TSS describes those management functions that may only be performed by the user and confirm that the TSS does not include an Administrator API for any of these management functions. This activity will be performed in conjunction with FMT_SMF_EXT.1.

Summary

Section 8.5.2 in the [ST] [\[1\]](#) describes the *Configuration Profiles*.

The evaluator verified that the functions 4, 5, 12, 13, 18 are only allowed to the user and not the administrator. The last paragraph of section 8.5.2 explains that configuration profiles can be deployed such that users are unable to override or remove restrictions set by the administrators or MDM administrators. Depending on the behavior defined in the configuration profile, users might be unable to perform or access management functions defined in table 5 of the [ST] [\[1\]](#).

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

FMT_MOF_EXT.1.2

TSS Assurance Activities

Assurance Activity AA-FMT_MOF_EXT.1.2-ASE-01

The evaluator shall verify that the TSS describes those management functions that may be performed by the Administrator, to include how the user is prevented from accessing, performing, or relaxing the function (if applicable), and how applications/APIs are prevented from modifying the Administrator configuration. The TSS also describes any functionality that is affected by administrator-configured policy and how. This activity will be performed in conjunction with FMT_SMF_EXT.1.

Summary

Section 8.5 of the [ST] [\[1\]](#) describes the *Specification of Management Functions (FMT)*.

Administrator configuration is setup via the configuration profiles deployed via the Apple Configurator tool, an email message, a web page, over-the-air configuration (As described in [OTA_CFG] [\[1\]](#)), and over-the-air using the MDM server. All settings or restrictions defined in these profiles restrict any options the user has. If the profiles are protected against removal by the administrator, the user cannot circumvent the restrictions or settings by removing the respective profile.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FMT_MOF_EXT.1.2-ATE-01

- **Test 1:** The evaluator shall use the test environment to deploy policies to Mobile Devices.
- **Test 2:** The evaluator shall create policies which collectively include all management functions which are controlled by the (enterprise) administrator and cannot be overridden/relaxed by the user as defined in FMT_MOF_EXT.1.2. The evaluator shall apply these policies to devices, attempt to override/relax each setting both as the user (if a setting is available) and as an application (if an API is available), and ensure that the TSF does not permit it. Note that the user may still apply a more restrictive policy than that of the administrator.
- **Test 3:** Additional testing of functions provided to the administrator are performed in conjunction with the testing activities for FMT_SMF_EXT.1.1.

Summary

The evaluator notes many previous test cases use the Apple Configurator 2 to deploy policies. All configuration options are tested by the Assurance Activities in FMT_SMF_EXT.1. For administrator-related configuration options, the Apple Configurator 2 is used.

2.1.5.2 Extended: Specification of Management Functions (FMT_SMF_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FMT_SMF_EXT.1-ASE-01

The evaluator shall verify that the TSS describes all management functions, what role(s) can perform each function, and how these functions are (or can be) restricted to the roles identified by FMT_MOF_EXT.1.

The following activities are organized according to the function number in the table. These activities include TSS assurance activities, AGD assurance activities, and test activities.

Summary

Section 8.5 in the [ST][\[1\]](#) describes the *Specification of Management Functions*.

Section 8.5 references to table 5 of the [ST][\[1\]](#) which describes which roles can perform which functions. The last paragraph of section 8.5.2 explains that configuration profiles can be deployed such that users are unable to override or remove restrictions set by the administrators or MDM administrators. Depending on the behavior defined in the configuration profile, users might be unable to perform or access management functions defined in table 5 of the [ST][\[1\]](#).

Assurance Activity AA-FMT_SMF_EXT.1-ASE-02

Function 1

The evaluator shall verify the TSS defines the allowable policy options: the range of values for both password length and lifetime, and a description of complexity to include character set and complexity policies (e.g., configuration and enforcement of number of uppercase, lowercase, and special characters per password).

Summary

Section 8.4 in the [ST][\[1\]](#) describes the *Identification and Authentication*.


Section 8.4 describes that the length, complexity and lifetime of the password is configurable by the user authenticated to the device or can also be set using a configuration profile if the device is under MDM.

Assurance Activity AA-FMT_SMF_EXT.1-ASE-03

Function 2

The evaluator shall verify the TSS defines the range of values for both timeout period and number of authentication failures for all supported authentication mechanisms.

Summary

Section 8.4 in the [ST]  describes the *Identification and Authentication*.

Section 8.4 describes that the maximum number of consecutive failed attempts to enter the right passcode can be set between 2 and 10, again by either the user or through a configuration profile; it also describes that the number of minutes for which the device can be idle before it locks itself automatically can be defined by the user or a configuration profile.

Assurance Activity AA-FMT_SMF_EXT.1-ASE-04

Function 4

The evaluator shall verify that the TSS includes a description of each radio and an indication of if the radio can be enabled/disabled along with what role can do so. In addition the evaluator shall verify that the frequency ranges at which each radio operates is included in the TSS.

Summary

Section 8.7.2 in the [ST]  describes the *Restricting Access to Wireless Networks*.

Section 8.7.2 describes how users and administrators can restrict the wireless networks on the TOE. This section references to Table 1: Devices Covered by the Evaluation and Table 2: Cellular Protocols Supported. The standards defined in these tables list the frequency ranges at which each radio operates.


Assurance Activity AA-FMT_SMF_EXT.1-ASE-05

Function 5

The evaluator shall verify that the TSS includes a description of each collection device and an indication of if it can be enabled/disabled along with what role can do so.

Summary

Section 8.5 in the [ST]  describes the *Specification of Management Functions (FMT)*.

Section 8.5 references to table 5 of the [ST]  which describes that the cameras can be enabled/disabled on a per-app basis by the user the administrator, and that the microphone can be /enabled/disabled on a per-app basis by the user only.


Assurance Activity AA-FMT_SMF_EXT.1-ASE-06

Function 8

The evaluator shall verify the TSS describes the allowable application installation policy options based on the selection included in the ST. If the application whitelist is selected, the evaluator shall verify that the TSS includes a description of each application characteristic upon which the whitelist may be based.

Summary

Section 8.5 in the [ST]  describes the *Specification of Management Functions (FMT)*.

Section 8.5 references to table 5 which defines that the administrator can set iOS to disallow any application to be installed. The application white-list option has not been selected in the [ST] .

Assurance Activity AA-FMT_SMF_EXT.1-ASE-07

Function 9 & Function 10

The evaluator shall verify that the TSS describes each category of keys/secrets that can be imported into the TSF's secure key storage.

Summary

Section 8.2.1 in the [ST]  describes the *Overview of Key Management*. Section 8.2.2 in the [ST]  describes the *Storage of Persistent Secrets and Private Keys by the Agent*.

Section 8.2.2 defines which keys/secrets can be imported and stored in the device, its purpose and how it is stored.


Assurance Activity AA-FMT_SMF_EXT.1-ASE-08

Function 12

The evaluator shall verify that the TSS describes each additional category of X.509 certificates and their use within the TSF.

Summary

The table in section 8.10 in the [ST]  describes the *Mapping to the Security Functional Requirements*.



The table entry FIA_X509_EXT.2 in section 8.10 specifies that X.509v3 certificate, and no other category has been selected in the [ST] .


Assurance Activity AA-FMT_SMF_EXT.1-ASE-09

Function 13

The evaluator shall examine the TSS to ensure that it contains a description of each management function that will be enforced by the enterprise once the device is enrolled.

Summary

Section 8.5.1 in the [ST]  describes the *Enrollment* management function. Section 8.5.2 in the [ST]  describes the *Configuration Profiles*.

Function 13 only selects one management function: Enroll the TOE in management. This is described in section 8.5.1 of the [ST] . All the management functions that will be enforced by the enterprise once the device is enrolled are described in section 8.5.2: these functions are enforced by the configuration profile installed on the device.

Assurance Activity AA-FMT_SMF_EXT.1-ASE-10

Function 14

The evaluator shall verify that the TSS includes an indication of what applications (e.g., user-installed applications, Administrator-installed applications, or Enterprise applications) can be removed along with what role can do so.

Summary

Section 8.5.2 in the [ST]  describes the *Configuration Profiles*.

Section 8.5.2 explains in the last paragraph that configuration profiles can define to unable users to override or remove restrictions set in place by an administrator (or MDM administrator). This includes installation or removal of application of the TOE.

Assurance Activity AA-FMT_SMF_EXT.1-ASE-12

Function 18

The evaluator shall ensure that the TSS includes a description of the Bluetooth profiles and services supported and the Bluetooth security modes and levels supported by the TOE. If function e is selected, the evaluator shall verify that the TSS describes any additional wireless technologies that may be used with Bluetooth, including Wi-Fi with Bluetooth High Speed and NFC as an Out of Band pairing mechanism. If function h is selected, the evaluator shall verify that all supported Bluetooth services are listed in the TSS as manageable and, if the TOE allows disabling by application rather than by service name, that a list of services for each application is also listed. If function i is selected, the evaluator shall verify that the TSS describes the method by which the level of security for pairings are managed, including whether the setting is performed for each pairing or is a global setting. If function j is selected, the evaluator shall verify that the TSS describes when Out of Band pairing methods are allowed and which ones are configurable.

Summary

Section 8.8.3 in the [ST]  describes the *Bluetooth* protocols.

In function 18, only functions a, b and c are selected. Section 8.8.3 describes the Bluetooth profiles and security modes and levels supported by the TOE.

Assurance Activity AA-FMT_SMF_EXT.1-ASE-13

Function 23

- **Test 23:** The evaluator shall verify that the TSS states if the TOE supports a BAF and/or hybrid authentication. If the TOE does not include a BAF and/or hybrid authentication this test is implicitly met.
 - a. [conditional] If a BAF is selected the evaluator shall verify that the TSS describes the procedure to enable/disable the BAF. If the TOE includes multiple BAFs, the evaluator shall verify that the TSS describes how to enable/disable each BAF, specifically if the different modalities can be individually enabled/disabled.
 - b. [conditional] If "Hybrid" is selected the evaluator shall verify that the TSS describes the procedure to enable/disable the hybrid (biometric credential and PIN/password) authentication.

Assurance Activity Note: It should be noted that the following functions are optional capabilities, if the function is implemented, then the following assurance activities shall be performed. The notation of "[conditional]" beside the function number indicates that if the function is not included in the ST, then there is no expectation that the assurance activity be performed.

Summary

Section 8.5.3 of the ST describes the *Biometric Authentication Factors (BAFs)*.

The TOE can support either Touch ID or Face ID as a BAF, but not both. The BAF can be set up in the security options in the TOE settings (Touch ID & Passcode menu OR Face ID & Passcode menu, depending on the device), to enable/disable Touch ID/Face ID and/or the passcode.

Assurance Activity AA-FMT_SMF_EXT.1-ASE-14

Function 24 [conditional]

The evaluator shall verify that the TSS includes a list of each externally accessible hardware port and an indication of if data transfer over that port can be enabled/disabled.

Summary

Function 24 is not selected in the [ST] , (Table 5).

Assurance Activity AA-FMT_SMF_EXT.1-ASE-15**Function 25 [conditional]**

The evaluator shall verify that the TSS describes how the TSF acts as a server in each of the protocols listed in the ST, and the reason for acting as a server.

Summary

Function 25 is not selected in the [ST] , (Table 5).

Assurance Activity AA-FMT_SMF_EXT.1-ASE-16**Function 29 [conditional]**

The evaluator shall verify that the TSS describes how approval for an application to perform the selected action (import, removal) with respect to certificates in the Trust Anchor Database is accomplished (e.g., a pop-up, policy setting, etc.).

Summary

Function 29 is not selected in the [ST] , (Table 5).

Assurance Activity AA-FMT_SMF_EXT.1-ASE-17**Function 31 [conditional]**

The evaluator shall ensure that the TSS describes which cellular protocols can be disabled.

Summary

Function 31 is not selected in the [ST] , (Table 5).

Assurance Activity AA-FMT_SMF_EXT.1-ASE-18**Function 34 [conditional]**

The evaluator shall verify that the TSS describes how the approval for exceptions for shared use of keys/secrets by multiple applications is accomplished (e.g., a pop-up, policy setting, etc.).

Summary

Function 34 is not selected in the [ST] , (Table 5).

Assurance Activity AA-FMT_SMF_EXT.1-ASE-19**Function 35 [conditional]**

The evaluator shall verify that the TSS describes how the approval for exceptions for destruction of keys/secrets by applications that did not import the key/secret is accomplished (e.g., a pop-up, policy setting, etc.).

Summary

Function 35 is not selected in the [ST] , (Table 5).

Assurance Activity AA-FMT_SMF_EXT.1-ASE-20

Function 36 [conditional]

The evaluator shall verify that the TSS describes any restrictions in banner settings (e.g., character limitations)

Summary

Section 8.7.3 in the [ST]  describes the *Lock Screen Banner Display*.

Section 8.7.3 describes that the banner can be set by displaying an image only when the screen is locked.

Assurance Activity AA-FMT_SMF_EXT.1-ASE-21

Function 39 [conditional]

The evaluator shall verify that the TSS includes a description of how data transfers can be managed over USB.

Summary

Function 39 is not selected in the [ST] , (Table 5).

Assurance Activity AA-FMT_SMF_EXT.1-ASE-22

Function 40 [conditional]

The evaluator shall verify that the TSS includes a description of available backup methods that can be enabled/disabled. If “selected applications or selected groups of applications are selected the TSS shall include which applications of groups of applications backup can be enabled/disabled.

Summary

Function 40 is not selected in the [ST] , (Table 5).

Assurance Activity AA-FMT_SMF_EXT.1-ASE-23

Function 41 [conditional]

The evaluator shall verify that the TSS includes a description of Hotspot functionality and USB tethering to include any authentication for these.

Summary

Function 41 is not selected in the [ST] , (Table 5).

Assurance Activity AA-FMT_SMF_EXT.1-ASE-24

Function 45 [conditional]

- **Test 45:** *The evaluator shall verify that the TSS contains guidance to configure the VPN as Always-On.*

Summary

Function 45 has not been selected by the ST author.

Assurance Activity AA-FMT_SMF_EXT.1-ASE-25

Function 46 [conditional]

- **Test 46:** *The evaluator shall verify that the TSS describes the procedure to revoke a biometric credential stored on the TOE.*

Summary

Function 46 is not selected in the [ST][\[1\]](#), (Table 5).

Assurance Activity AA-FMT_SMF_EXT.1-ASE-26

Function 47

The evaluator shall verify that the TSS describes all assigned security management functions and their intended behavior.

Summary

Function has not been selected by the ST author.

Guidance Assurance Activities

Assurance Activity AA-FMT_SMF_EXT.1-AGD-01

The evaluator shall consult the AGD guidance to perform each of the specified tests, iterating each test as necessary if both the user and administrator may perform the function. The evaluator shall verify that the AGD guidance describes how to perform each management function, including any configuration details. For each specified management function tested, the evaluator shall confirm that the underlying mechanism exhibits the configured setting.

Summary

Instructions to perform each management function is provided throughout [CCGUIDE][\[1\]](#) which has been assessed by the evaluator while performing other guidance assurance activities. Additionally, [CCGUIDE][\[1\]](#) section 3.5 *Security Management Configuration*, the provided guidance column in Table 9 "Management Functions" references the corresponding section(s) in [CCGUIDE][\[1\]](#) where guidance can be found to perform the respective management function.

Assurance Activity AA-FMT_SMF_EXT.1-AGD-02

Function 4

[The evaluator shall verify that the TSS includes a description of each radio and an indication of if the radio can be enabled/disabled along with what role can do so.]

The evaluator shall confirm that the AGD guidance describes how to perform the enable/disable function for each radio.

Summary

[CCGUIDE][\[1\]](#) refers to the user guides [iPhone_UG][\[1\]](#) and [iPad_UG][\[1\]](#) for description of instructions on how to perform the enable/disable function of the devices. For example, section 3.4.4 *Bluetooth Configuration* of [CCGUIDE][\[1\]](#) refers to [iPhone_UG][\[1\]](#) and [iPad_UG][\[1\]](#) section *iPhone and other devices* and *Connect Bluetooth devices*, respectively, for instructions how to turn Bluetooth on and off and how to pair and unpair a Bluetooth device.

Assurance Activity AA-FMT_SMF_EXT.1-AGD-03

Function 5

[The evaluator shall verify that the TSS includes a description of each collection device and an indication of if it can be enabled/disabled along with what role can do so.] The evaluator shall confirm that the AGD guidance describes how to perform the enable/disable function.

Summary

Instructions to perform each management function (including enable/disable the function) is provided throughout [CCGUIDE] [\[1\]](#) which has been assessed by the evaluator while performing other guidance assurance activities. In particular, [CCGUIDE] [\[1\]](#) Table 9 "Management Functions" lists for each management function claimed in Table 5 of ST, the corresponding section(s) in the [CCGUIDE] [\[1\]](#) where guidance is provided.

Assurance Activity AA-FMT_SMF_EXT.1-AGD-04**Function 11**

The evaluator shall review the AGD guidance to determine that it describes the steps needed to import, modify, or remove certificates in the Trust Anchor database, and that the users that have authority to import those certificates (e.g., only administrator, or both administrators and users) are identified.

Summary

[CCGUIDE] [\[1\]](#) section 3.2.6 *Keys/Secrets Import/Destruction* describes import, modification, and destruction of keys, secrets, and/or certificates. It states the following:

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] [\[1\]](#) and the "Keychain Services Programming Guide" [KEYCHAINPG] [\[1\]](#), 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.

Assurance Activity AA-FMT_SMF_EXT.1-AGD-05**Function 13**

[The evaluator shall examine the TSS to ensure that it contains a description of each management function that will be enforced by the enterprise once the device is enrolled.] The evaluator shall examine the AGD guidance to determine that this same information is present.

Summary

Table 9 of section 3.5 *Security Management Configuration* of [CCGUIDE] [\[1\]](#) outlining all management functions and policies that can be performed and enforced both by the user and the administrator. One additional column of this table also specifies if the function or policy can be performed or enforced by the enterprise when the device is enrolled. Corresponding guidance for each respective management function when enrolled is provided in the right-most column "Provided Guidance" of Table 9.

Assurance Activity AA-FMT_SMF_EXT.1-AGD-06**Function 14**

The evaluator shall examine the AGD guidance to determine that it details, for each type of application that can be removed, the procedures necessary to remove those applications and their associated data. For the purposes of this assurance activity, "associated data" refers to data that are created by the app during its operation that do not exist independent of the app's existence, for instance, configuration data, or e-mail information that's part of an e-mail client. It does not, on the other hand, refer to data such as word processing documents (for a word processing app) or photos (for a photo or camera app).

Summary

[CCGUIDE] [\[1\]](#) section 3.5.1 *Install/Remove Apps from the TOE* provides information about removing apps. It states the following:

The Administrator can install apps to the TOE using an MDM system or Apple Configurator. Refer to [iOSDeployRef] [\[1\]](#) section *app and book distribution* and 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] [\[1\]](#).

Users can install or remove an app from their TOE device. See the [iPhone_UG] [\[1\]](#) and the [iPad_UG] [\[1\]](#) *App Store>> section Purchase, redeem, and download*.

Assurance Activity AA-FMT_SMF_EXT.1-AGD-08

Function 19

The evaluator shall examine the AGD Guidance to determine that it specifies, for at least each category of information selected for Function 21, how to enable and disable display information for that type of information in the locked state.

Summary

[CCGUIDE] [\[1\]](#) section 3.5.2 *Configure Access and Notification in Locked State* states that access to certain optional features can be allowed when the TOE device is in locked state. These optional features include:

- Email notification
- Calendar appointment
- Text message notification

This section provides guidance (which refers to [iPhone_UG] [\[1\]](#) and [iPad_UG] [\[1\]](#)) on how to enable/disable display certain notifications when the TOE device is locked.

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>>Face ID & 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.

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] [\[1\]](#) and the [iPhone_UG] [\[1\]](#) *Basics section Notification*.

Alternatively, displaying notifications when in the locked state can be allowed/prohibited via the allowLockScreenNotificationsView key in the Restrictions Payload of a Configuration Profile which is described in more detailed in [IOS_CFG] [\[1\]](#).

Assurance Activity AA-FMT_SMF_EXT.1-AGD-09

Function 24 [conditional]

[The evaluator shall verify that the TSS includes a list of each externally accessible hardware port and an indication of if data transfer over that port can be enabled/disabled.] AGD guidance will describe how to perform the enable/disable function.

Summary

This function is optional and was not selected in [ST].

Assurance Activity AA-FMT_SMF_EXT.1-AGD-10**Function 27 [conditional]**

The evaluator shall examine the AGD guidance to determine that it describes how to enable and disable any "Forgot Password", password hint, or remote authentication (to bypass local authentication mechanisms) capability.

Summary

The TOE does not support the enablement/disablement of "Forgot Password", password hint, or remote authentication (to bypass local authentication mechanisms) capability.

Assurance Activity AA-FMT_SMF_EXT.1-AGD-11**Function 29 [conditional]**

The evaluator shall also verify that the API documentation provided according to Section 5.2.2 includes any security functions (import, modification, or destruction of the Trust Anchor Database) allowed by applications.

Summary

[CCGUIDE] section 3.4.6 X.509 Certificate Configuration states the following:

- 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.
- Administrators can view all certificates on a device and remove any certificates that have been 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.

Assurance Activity AA-FMT_SMF_EXT.1-AGD-12**Function 31 [conditional]**

The evaluator shall confirm that the AGD guidance describes the procedure for disabling each cellular protocol identified in the TSS.

Summary

This function is optional and was not selected in [ST].

Test Activities

Assurance Activity AA-FMT_SMF_EXT.1-ATE-01

Test activities specified below shall take place in the test environment described in the Assurance Activity for FPT_TUD_EXT.1.1, FPT_TUD_EXT.1.2, and FPT_TUD_EXT.1.3. The evaluator shall consult the AGD guidance to perform each of the specified tests, iterating each test as necessary if both the user and administrator may perform the function. The evaluator shall verify that the AGD guidance describes how to perform each management function, including any configuration details. For each specified management function tested, the evaluator shall confirm that the underlying mechanism exhibits the configured setting.

Summary

The evaluator performed the subsequent FMT_SMF_EXT.1 tests in the same testing environment as previous testing and tested using the evaluated configuration.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-02

Function 1

- **Test 1:** *The evaluator shall exercise the TSF configuration as the administrator and perform positive and negative tests, with at least two values set for each variable setting, for each of the following:*
 - *minimum password length*
 - *minimum password complexity*
 - *maximum password lifetime*

Summary

The evaluator used the Apple Configurator 2 to set options for password quality requirements and verified that configuration settings for password length, quality, and expiration were enforced properly.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-03

Function 2

- **Test 2:** *The evaluator shall exercise the TSF configuration as the administrator. The evaluator shall perform positive and negative tests, with at least two values set for each variable setting, for each of the following.*
 - *screen-lock enabled/disabled*
 - *screen lock timeout*
 - *number of authentication failures (may be combined with test for FIA_AFL_EXT.1)*

Summary

The evaluator used the Apple Configurator 2 to set options for auto lock and verified that the configured timeout is used to lock the device. The evaluator also verified that the user was not able to manually set the auto lock timeout to a higher (longer) timeout value.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-04

Function 3

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

- [conditional] If “per-app basis” is selected, the evaluator shall create two applications and enable one to use the VPN and the other to not use the VPN. The evaluator shall exercise each application (attempting to access network resources; for example, by browsing different websites) individually while capturing packets from the TOE. The evaluator shall verify from the packet capture that the traffic from the VPN-enabled application is encapsulated in IPsec and that the traffic from the VPN-disabled application is not encapsulated in IPsec.
- [conditional] If “per-groups of application basis” is selected, the evaluator shall create two applications and the applications shall be placed into different groups. Enable one application group to use the VPN and the other to not use the VPN. The evaluator shall exercise each application (attempting to access network resources; for example, by browsing different websites) individually while capturing packets from the TOE. The evaluator shall verify from the packet capture that the traffic from the application in the VPN-enabled group is encapsulated in IPsec and that the traffic from the application in the VPN-disabled group is not encapsulated in IPsec.

Summary

Test 3a is tested by the Assurance Activities in FDP_IFC_EXT.1. Tests 3b and 3c are not applicable to the TOE.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-05

Function 4

The evaluator shall ensure that minimal signal leakage enters the RF shielded enclosure (i.e, Faraday bag, Faraday box, RF shielded room) by performing the following steps:

- Step 1: Place the antenna of the spectrum analyzer inside the RF shielded enclosure.
- Step 2: Enable “Max Hold” on the spectrum analyzer and perform a spectrum sweep of the frequency range between 300MHz – 6000MHz, in 1 KHz steps (this range should encompass 802.11, 802.15, GSM, UMTS, LTE and GPS). This range will not address NFC 13.56MHz, another test should be set up with similar constraints to address NFC.

If power above -90 dBm is observed, the Faraday box has too great of signal leakage and shall not be used to complete the test for Function 4.

- **Test 4:** The evaluator shall exercise the TSF configuration as all roles specified in the TSS to enable and disable the state of each radio (e.g. Wi-Fi, GPS, cellular, NFC, Bluetooth). Additionally, the evaluator shall repeat the steps below, booting into any auxiliary boot mode supported by the device. For each radio, the evaluator shall:
 - Step 1: Place the antenna of the spectrum analyzer inside the RF shielded enclosure. Configure the spectrum analyzer to sweep desired frequency range for the radio to be tested (based on range provided in the TSS)). The ambient noise floor shall be set to - 110dBm. Place the TOE into the RF shielded enclosure to isolate them from all other RF traffic.
 - Step 2: The evaluator shall create a baseline of the expected behavior of RF signals. The evaluator shall power on the device, ensure the radio in question is enabled, power off the device, enable “Max Hold” on the spectrum analyzer and power on the device. The evaluator shall wait 2 minutes at each Authentication Factor interface prior to entering the necessary password to complete the boot process, waiting 5 minutes after the device is fully booted. The evaluator shall observe that RF spikes are present at the expected uplink channel frequency. The evaluator shall clear the “Max Hold” on the spectrum analyzer.
 - Step 3: The evaluator shall verify the absence of RF activity for the uplink channel when the radio in question is disabled. The evaluator shall complete the following test five times. The evaluator shall power on the device, ensure the radio in question is disabled, power off the device, enable “Max Hold” on the spectrum analyzer and power on the device. The evaluator shall wait 2 minutes at each Authentication Factor interface prior to entering the necessary password to complete the boot process, waiting 5 minutes after the device is fully booted. The evaluator shall clear the “Max Hold” on the spectrum analyzer. If a spike of RF activity for the uplink channel of the specific radio frequency band is observed at any time (either at an Authentication Factor interface or when the device is fully booted) it is deemed that the radio is enabled.

Summary

The evaluator scanned a variety of well-known frequencies as described while in a shielded room and found no unexpected spikes in the signals emitted by the TOE device.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-06

Function 5

- **Test 5:** The evaluator shall perform the following test(s):
 - a. The evaluator shall exercise the TSF configuration as the administrator and, if not restricted to the administrator, the user, to enable and disable the state of each audio or visual collection devices (e.g. camera, microphone) listed by the ST author. For each collection device, the evaluator shall disable the device and then attempt to use its functionality. The evaluator shall reboot the TOE and verify that disabled collection devices may not be used during or early in the boot process. Additionally, the evaluator shall boot the device into each available auxiliary boot mode and verify that the collection device cannot be used.
 - b. [conditional] If “per-app basis” is selected, the evaluator shall create two applications and enable one to use access the A/V device and the other to not access the A/V device. The evaluator shall exercise each application attempting to access the A/V device individually. The evaluator shall verify that the enabled application is able to access the A/V device and the disabled application is not able to access the A/V device.
 - c. [conditional] If “per-groups of application basis” is selected, the evaluator shall create two applications and the applications shall be placed into different groups. Enable one group to access the A/V device and the other to not access the A/V device. The evaluator shall exercise each application attempting to access the A/V device individually. The evaluator shall verify that the application in the enabled group is able to access the A/V device and the application in the disabled group is not able to access the A/V device.

Summary

For test 5a, the evaluator verified that both cameras and microphones (on a per app basis) can be disabled and are not usable. He also booted the device to an auxiliary boot mode and verified that both cameras and microphones were still not usable.

For test 5b, the evaluator disabled both the cameras and microphones for use by a specific application and verified that they were then inaccessible to the application.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-07**Function 6**

- **Test 6:** The evaluator shall use the test environment to instruct the TSF, both as a user and as the administrator, to command the device to transition to a locked state, and verify that the device transitions to the locked state upon command.

Summary

The evaluator used both local and remote locking of the TOE and verified in each case that the device was locked.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-08**Function 7**

- **Test 7:** The evaluator shall use the test environment to instruct the TSF, both as a user and as the administrator, to command the device to perform a wipe of protected data. The evaluator must ensure that this management setup is used when conducting the assurance activities in FCS_CKM_EXT.5.

Summary

The evaluator triggered a wipe of all data via the device UI by selecting the "Erase All Content and Settings" option under Settings, General, Reset.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-09**Function 8**

- **Test 8:** *The evaluator shall exercise the TSF configuration as the administrator to restrict particular applications, sources of applications, or application installation according to the AGD guidance. The evaluator shall attempt to install unauthorized applications and ensure that this is not possible. The evaluator shall, in conjunction, perform the following specific tests:*
 - a. *Test 8a: [conditional] The evaluator shall attempt to connect to an unauthorized repository in order to install applications.*
 - b. *Test 8b: [conditional] The evaluator shall attempt to install two applications (one whitelisted, and one not) from a known allowed repository and verify that the application not on the whitelist is rejected. The evaluator shall also attempt to side-load executables or installation packages via USB connections to determine that the white list is still adhered to.*

Summary

The evaluator disabled installation of applications using a policy deployed with the Apple Configurator 2. He then tried to install an application from the App Store but was unable to do so.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-11

Function 9 & Function 10

- **Test 9 & Test 10:** *The test of these functions is performed in association with FCS_STG_EXT.1.*

Summary

Testing of these functions is performed in association with the Assurance Activities of FCS_STG_EXT.1.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-12

Function 11

- **Test 11:** *The evaluator shall import certificates according to the AGD guidance as the user and/or as the administrator, as determined by the administrative guidance. The evaluator shall verify that no errors occur during import. The evaluator should perform an action requiring use of the X.509v3 certificate to provide assurance that installation was completed properly.*

Summary

Testing of these functions is performed in association with the Assurance Activities of FCS_TLSC_EXT.1 (for both MDFPP and WLAN).

Assurance Activity AA-FMT_SMF_EXT.1-ATE-13

Function 12

- **Test 12:** *The evaluator shall remove an administrator-imported certificate and any other categories of certificates included in the assignment of function 14 from the Trust Anchor Database according to the AGD guidance as the user and as the administrator.*

Summary

The evaluator added and removed certificates with the Apple Configurator 2 numerous times while performing the Assurance Activities of FIA_X509_EXT.1 and confirmed the certificates were removed.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-14

Function 13

- **Test 13:** *The evaluator shall verify that user approval is required to enroll the device into management.*

Summary

The evaluator performed all steps necessary to enroll a device into management. One of these early steps requires the device to be registered with the Apple Profile Manager on the managing device. The evaluator determined this step constitutes user approval for enrolling the device.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-15

Function 14

- **Test 14:** *The evaluator shall attempt to remove applications according to the AGD guidance and verify that the TOE no longer permits users to access those applications or their associated data.*

Summary

The evaluator removed an application and verified the application was no longer available.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-16

Function 15

- **Test 15:** *The evaluator shall attempt to update the TSF system software following the procedures in the AGD guidance and verify that updates correctly install and that the version numbers of the system software increase.*

Summary

The evaluator performed an iOS update and verified that it installed correctly and the version number was updated.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-17

Function 16

- **Test 16:** *The evaluator shall attempt to install an application following the procedures in the AGD guidance and verify that the application is installed and available on the TOE.*

Summary

The evaluator installed an application from the App Store and was then able to use the application.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-18

Function 17

- **Test 17:** *The evaluator shall attempt to remove any Enterprise applications from the device by following the administrator guidance. The evaluator shall verify that the TOE no longer permits users to access those applications or their associated data.*

Summary

The evaluator used the Apple Configurator 2 to deploy an application to the TOE and launch it to verify that it runs correctly. He then used the Apple Configurator 2 to remove the application and verified that it was no longer available on the TOE.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-19

Function 18

- **Test 18:** The evaluator shall use a Bluetooth-specific protocol analyzer to perform the following tests of each sub-function:
 - a. The evaluator shall disable the Discoverable mode and shall verify that other Bluetooth BR/EDR devices cannot detect the TOE. The evaluator shall use the protocol analyzer to verify that the TOE does not respond to inquiries from other devices searching for Bluetooth devices. The evaluator shall enable Discoverable mode and verify that other devices can detect the TOE and that the TOE sends response packets to inquiries from searching devices.
 - b. The evaluator shall examine Bluetooth traffic from the TOE to determine the current Bluetooth device name, change the Bluetooth device name, and verify that the Bluetooth traffic from the TOE lists the new name.
 - c. [conditional] The evaluator shall examine Bluetooth traffic from the TOE to determine the current Bluetooth device name for BR/EDR and LE. The evaluator shall change the Bluetooth device name for LE independently of the device name for BR/EDR. The evaluator shall verify that the Bluetooth traffic from the TOE lists the new name.
 - d. [conditional] The evaluator shall disable Bluetooth BR/EDR and enable Bluetooth LE. The evaluator shall examine Bluetooth traffic from the TOE to confirm that only Bluetooth LE traffic is present. The evaluator shall repeat the test with Bluetooth BR/EDR enabled and Bluetooth LE disabled, confirming that only Bluetooth BR/EDR is present.
 - e. [conditional] The evaluator shall disable additional wireless technologies for the TOE and verify that Bluetooth High Speed is not able to send Bluetooth traffic over Wi-Fi, and that NFC cannot be used for pairing. The evaluator shall enable additional wireless technologies and verify that Bluetooth High Speed uses Wi-Fi or that the device can pair using NFC.
 - f. [conditional] The evaluator shall enable Advertising for Bluetooth LE, verify that the advertisements are captured by the protocol analyzer, disable Advertising, and verify that no advertisements from the device are captured by the protocol analyzer.
 - g. [conditional] The evaluator shall enable Connectable mode and verify that other Bluetooth devices may pair with the TOE and (if the devices were bonded) re-connect after pairing and disconnection. For BR/EDR devices: The evaluator shall use the protocol analyzer to verify that the TOE responds to pages from the other devices and permits pairing and re-connection. The evaluator shall disable Connectable mode and verify that the TOE does not respond to pages from remote Bluetooth devices, thereby not permitting pairing or re-connection. For LE: The evaluator shall use the protocol analyzer to verify that the TOE sends connectable advertising events and responds to connection requests. The evaluator shall disable Connectable mode and verify that the TOE stops sending connectable advertising events and stops responding to connection requests from remote Bluetooth devices.
 - h. [conditional] The evaluator shall verify that all supported Bluetooth services and/or profiles are listed in the TSS as manageable and, if the TOE allows disabling by application rather than by service and/or profile name, that a list of services and/or profile for each application is also listed.
 - i. [conditional] The evaluator shall allow low security modes/levels on the TOE and shall initiate pairing with the TOE from a remote device that allows only something other than Security Mode 4/Level 3 or Security Mode 4/Level 4 (for BR/EDR), or Security Mode 1/Level 3 (for LE). (For example, a remote BR/EDR device may claim Input/Output capability "NoInputNoOutput" and state that man-in-the-middle (MiTM) protection is not required. A remote LE device may not support encryption.) The evaluator shall verify that this pairing attempt succeeds due to the TOE falling back to the low security mode/level. The evaluator shall then remove the pairing of the two devices, prohibit the use of low security modes/levels on the TOE, then attempt the connection again. The evaluator shall verify that the pairing attempt fails. With the low security modes/levels disabled, the evaluator shall initiate pairing from the TOE to a remote device that supports Security Mode 4/Level 3 or Security Mode 4/Level 4 (for BR/EDR) or Security Mode 1/Level 3 (for LE). The evaluator shall verify that this pairing is successful and uses the high security mode/level.
 - j. [conditional] The evaluator shall attempt to pair using each of the Out of Band pairing methods, verify that the pairing method works, iteratively disable each pairing method, and verify that the pairing method fails.

Summary

For the first three tests (a, b, and c), the evaluator obtained the device ID using the hcitool scan tool with the TOE in Bluetooth pairing mode. He then removed the TOE from Bluetooth pairing mode and verified that the scanner did not see the TOE as an available Bluetooth device. He then scanned normal Bluetooth traffic from the TOE to verify it includes the device name, changed the device name, and verified that the new name then appears in the Bluetooth traffic. The evaluator also disabled encryption on the device to which the TOE was paired and verified the connection was terminated by the TOE.

The remaining conditional tests (d through j) are not applicable to the TOE because those functions are not claimed in [ST].

Assurance Activity AA-FMT_SMF_EXT.1-ATE-20

Function 19

- **Test 19:** For each category of information listed in the AGD guidance, the evaluator shall verify that when that TSF is configured to limit the information according to the AGD, the information is no longer displayed in the locked state.

Summary

The evaluator configured the TOE to disable notifications via the Apple Configurator 2 and verified that no notifications are shown when the device is locked.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-21

Function 20

- **Test 20:** The evaluator shall exercise the TSF configuration as the administrator and, if not restricted to the administrator, the user, to enable system-wide data-at-rest protection according to the AGD guidance. The evaluator shall ensure that all assurance activities for FDP_DAR are conducted with the device in this configuration.

Summary

Function not included in [ST].

Assurance Activity AA-FMT_SMF_EXT.1-ATE-22

Function 21

- **Test 21:** The evaluator shall exercise the TSF configuration as the administrator and, if not restricted to the administrator, the user, to enable removable media's data-at-rest protection according to the AGD guidance. The evaluator shall ensure that all assurance activities for DAR (see FDP_DAR) are conducted with the device in this configuration.

Summary

Function not included in [ST].

Assurance Activity AA-FMT_SMF_EXT.1-ATE-23

Function 22

- **Test 22:** The evaluator shall perform the following tests.
 - a. The evaluator shall enable location services device-wide and shall verify that an application (such as a mapping application) is able to access the TOE's location information. The evaluator shall disable location services device-wide and shall verify that an application (such as a mapping application) is unable to access the TOE's location information.
 - b. [conditional] If "per-app basis" is selected, the evaluator shall create two applications and enable one to use access the location services and the other to not access the location services. The evaluator shall exercise each application attempting to access location services individually. The evaluator shall verify that the enabled application is able to access the location services and the disabled application is not able to access the location services.

Summary

The evaluator verified that the TOE location is enabled and can be obtained using Maps. He then disabled Location Services for the device and verified the TOE's location was no longer shown in Maps. He restored Location Services for the device, verified it worked again, and then disabled Location Services for the Maps app only, and verified that the location was not shown.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-24

Function 23

- **Test 23:** *The evaluator shall verify that the TSS states if the TOE supports a BAF and/or hybrid authentication. If the TOE does not include a BAF and/or hybrid authentication this test is implicitly met.*
 - a. *[conditional] If a BAF is selected the evaluator shall verify that the TSS describes the procedure to enable/disable the BAF. If the TOE includes multiple BAFs, the evaluator shall verify that the TSS describes how to enable/disable each BAF, specifically if the different modalities can be individually enabled/disabled. The evaluator shall configure the TOE to allow each supported BAF to authenticate and verify that successful authentication can be achieved using the BAF. The evaluator shall configure the TOE to disable the use of each supported BAF for authentication and confirm that the BAF cannot be used to authenticate.*
 - b. *[conditional] If "Hybrid" is selected the evaluator shall verify that the TSS describes the procedure to enable/disable the hybrid (biometric credential and PIN/password) authentication. The evaluator shall configure the TOE to allow hybrid authentication to authenticate and confirm that successful authentication can be achieved using the hybrid authentication. The evaluator shall configure the TOE to disable the use of hybrid authentication and confirm that the hybrid authentication cannot be used to authenticate.*

Summary

Function not included in [ST].

Assurance Activity AA-FMT_SMF_EXT.1-ATE-25

Assurance Activity Note: *It should be noted that the following functions are optional capabilities, if the function is implemented, then the following assurance activities shall be performed. The notation of "[conditional]" beside the function number indicates that if the function is not included in the ST, then there is no expectation that the assurance activity be performed.*

Function 24 [conditional]

- **Test 24:** *The evaluator shall exercise the TSF configuration to enable and disable data transfer capabilities over each externally accessible hardware ports (e.g. USB, SD card, HDMI) listed by the ST author. The evaluator shall use test equipment for the particular interface to ensure that no low-level signaling is occurring on all pins used for data transfer when they are disabled. For each disabled data transfer capability, the evaluator shall repeat this test by rebooting the device into the normal operational mode and verifying that the capability is disabled throughout the boot and early execution stage of the device.*

Summary

The evaluator successfully enabled and disabled the biometric authentication factor appropriate for each device.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-27

Function 25 [conditional]

- **Test 25:** *The evaluator shall attempt to disable each listed protocol in the assignment. The evaluator shall verify that remote devices can no longer access the TOE or TOE resources using any disabled protocols.*

Summary

Function not included in [ST].

Assurance Activity AA-FMT_SMF_EXT.1-ATE-28

Function 26 [conditional]

- **Test 26:** The evaluator shall exercise the TSF configuration as the administrator and, if not restricted to the administrator, the user, to enable and disable any developer mode. The evaluator shall test that developer mode access is not available when its configuration is disabled. The evaluator shall verify the developer mode remains disabled during device reboot.

Summary

Function not included in [ST].

Assurance Activity AA-FMT_SMF_EXT.1-ATE-29**Function 27 [conditional]**

- **Test 27:** For each mechanism listed in the AGD guidance that provides a “Forgot Password” feature or other means where the local authentication process can be bypassed, the evaluator shall disable the feature and ensure that they are not able to bypass the local authentication process.

Summary

Function not included in [ST].

Assurance Activity AA-FMT_SMF_EXT.1-ATE-30**Function 28 [conditional]**

- **Test 28:** The evaluator shall attempt to wipe Enterprise data resident on the device according to the administrator guidance. The evaluator shall verify that the data is no longer accessible by the user.

Summary

The evaluator notes there is no difference between enterprise applications and their data and regular applications and their data, thus the test for Function 16 also covers this test. The Assurance Activities associated with FCS_CKM_EXT.4 demonstrate that all data is wiped on the device.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-31**Function 29 [conditional]**

- **Test 29:** The evaluator shall perform one of the following tests:
 - a. [conditional] If applications may import certificates to the Trust Anchor Database, the evaluator shall write, or the developer shall provide access to, an application that imports a certificate into the Trust Anchor Database. The evaluator shall verify that the TOE requires approval before allowing the application to import the certificate:
 - The evaluator shall deny the approvals to verify that the application is not able to import the certificate. Failure of import shall be tested by attempting to validate a certificate that chains to the certificate whose import was attempted (as described in the Assurance Activity for FIA_X509_EXT.1).
 - The evaluator shall repeat the test, allowing the approval to verify that the application is able to import the certificate and that validation occurs.
 - b. [conditional] If applications may remove certificates in the Trust Anchor Database, the evaluator shall write, or the developer shall provide access to, an application that removes certificates from the Trust Anchor Database. The evaluator shall verify that the TOE requires approval before allowing the application to remove the certificate:
 - The evaluator shall deny the approvals to verify that the application is not able to remove the certificate. Failure of removal shall be tested by attempting to validate a certificate that chains to the certificate whose removal was attempted (as described in the Assurance Activity for FIA_X509_EXT.1).

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

Summary

Function not included in [ST] .

Assurance Activity AA-FMT_SMF_EXT.1-ATE-32

Function 30 [conditional]

- **Test 30:** *The test of this function is performed in conjunction with FIA_X509_EXT.2.2.*

Summary

Testing of this function is performed in association with the Assurance Activities of FIA_X509_EXT.2.2.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-33

Function 31 [conditional]

- **Test 31:** *The evaluator shall attempt to disable each cellular protocol according to the administrator guidance. The evaluator shall attempt to connect the device to a cellular network and, using network analysis tools, verify that the device does not allow negotiation of the disabled protocols.*

Summary

Function not included in [ST] .

Assurance Activity AA-FMT_SMF_EXT.1-ATE-34

Function 32 [conditional]

- **Test 32:** *The evaluator shall attempt to read any device audit logs according to the administrator guidance and verify that the logs may be read. This test may be performed in conjunction with the assurance activity of FAU_GEN.1.*

Summary

Function not included in [ST] .

Assurance Activity AA-FMT_SMF_EXT.1-ATE-35

Function 33 [conditional]

- **Test 33:** *The test of this function is performed in conjunction with FPT_TUD_EXT.2.5.*

Summary

The test of this function is performed in conjunction with FPT_TUD_EXT.2.5.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-36

Function 34 [conditional]

- **Test 34:** *The test of this function is performed in conjunction with FPT_TUD_EXT.4.1.*

Summary

Function not included in [ST].

Assurance Activity AA-FMT_SMF_EXT.1-ATE-37

Function 35 [conditional]

- **Test 35:** The test of this function is performed in conjunction with FCS_STG_EXT.1.

Summary

Function not included in [ST].

Assurance Activity AA-FMT_SMF_EXT.1-ATE-38

Function 36 [conditional]

- **Test 36:** The test of this function is performed in conjunction with FTA_TAB.1.

Summary

The test of this function is performed in conjunction with FTA_TAB.1

Assurance Activity AA-FMT_SMF_EXT.1-ATE-39

Function 37 [conditional]

- **Test 37:** The test of this function is performed in conjunction with FAU_SEL.1.

Summary

The test of this function is performed in conjunction with FAU_SEL.1.

Assurance Activity AA-FMT_SMF_EXT.1-ATE-40

Function 38 [conditional]

- **Test 38:** The test of this function is performed in conjunction with FPT_NOT_EXT.1.2.

Summary

Function not included in [ST].

Assurance Activity AA-FMT_SMF_EXT.1-ATE-41

Function 39 [conditional]

- **Test 39:** The evaluator shall perform the following tests based on the selections made in the table:
 - a. [conditional] The evaluator shall disable USB mass storage mode, attach the device to a computer, and verify that the computer cannot mount the TOE as a drive. The evaluator shall reboot the TOE and repeat this test with other supported auxiliary boot modes.
 - b. [conditional] The evaluator shall disable USB data transfer without user authentication, attach the device to a computer, and verify that the TOE requires user authentication before the computer can access TOE data. The evaluator shall reboot the TOE and repeat this test with other supported auxiliary boot modes.
 - c. [conditional] The evaluator shall disable USB data transfer without connecting system authentication, attach the device to a computer, and verify that the TOE requires connecting system authentication before the computer can access TOE data. The evaluator shall then connect the TOE to another computer and verify that the computer cannot access TOE data. The evaluator shall then connect the TOE to the original computer and verify that the computer can access TOE data.

Summary

Function not included in [ST] .

Assurance Activity AA-FMT_SMF_EXT.1-ATE-42

Function 40 [conditional]

- **Test 40:** If “all applications” is selected, the evaluator shall disable each selected backup location in turn and verify that the TOE cannot complete a backup. The evaluator shall then enable each selected backup location in turn and verify that the TOE can perform a backup.
If “selected applications” is selected, the evaluator shall disable each selected backup location in turn and verify that for the selected application the TOE prevents backup from occurring. The evaluator shall then enable each selected backup location in turn and verify that for the selected application the TOE can perform a backup.
If “selected groups of applications” is selected, the evaluator shall disable each selected backup location in turn and verify that for a group of applications the TOE prevents the backup from occurring. The evaluator shall then enable each selected backup location in turn and verify for the group of application the TOE can perform a backup.
If “configuration data” is selected, the evaluator shall disable each selected backup location in turn and verify that the TOE prevents the backup of configuration data from occurring. The evaluator shall then enable each selected backup location in turn and verify that the TOE can perform a backup of configuration data.

Summary

Function not included in [ST] .

Assurance Activity AA-FMT_SMF_EXT.1-ATE-43

Function 41 [conditional]

- **Test 41:** The evaluator shall perform the following tests based on the selections in 0.
 - a. [conditional] The evaluator shall enable hotspot functionality with each of the of the support authentication methods. The evaluator shall connect to the hotspot with another device and verify that the hotspot functionality requires the configured authentication method.
 - b. [conditional] The evaluator shall enable USB tethering functionality with each of the of the support authentication methods. The evaluator shall connect to the TOE over USB with another device and verify that the tethering functionality requires the configured authentication method.

Summary

Function not included in [ST] .

Assurance Activity AA-FMT_SMF_EXT.1-ATE-44

Function 42 [conditional]

- **Test 42:** The test of this function is performed in conjunction with FDP_ACF_EXT.1.2.

Summary

Function not included in [ST] .

Assurance Activity AA-FMT_SMF_EXT.1-ATE-45

Function 43 [conditional]

- **Test 43:** The evaluator shall set a policy to cause a designated application to be placed into a particular application group. The evaluator shall then install the designated application and verify that it was placed into the correct group.

Summary

Function not included in [ST] .

Assurance Activity AA-FMT_SMF_EXT.1-ATE-46

Function 44 [conditional]

- **Test 44:** The evaluator shall attempt to unenroll the device from management and verify that the steps described in FMT_SMF_EXT.2.1 are performed. This test should be performed in conjunction with the FMT_SMF_EXT.2.1 assurance activity.

Summary

Function not included in [ST] .

Assurance Activity AA-FMT_SMF_EXT.1-ATE-47

Function 45 [conditional]

- **Test 45:** The evaluator shall configure the VPN as Always-On and perform the following test.
 - a. The evaluator shall verify that when the VPN is connected all traffic is routed through the VPN. This test is performed in conjunction with FDP_IFC_EXT.1.1.
 - b. The evaluator shall verify that when the VPN is not established, that no traffic leaves the device. The evaluator shall ensure that the TOE has network connectivity and that the VPN is established. The evaluator shall use a packet sniffing tool to capture the traffic leaving the TOE. The evaluator shall disable the VPN connection on the server side. The evaluator shall perform actions with the device such as navigating to websites, using provided applications, and accessing other Internet resources and verify that no traffic leaves the device.
 - c. The evaluator shall verify that the TOE has network connectivity and that the VPN is established. The evaluator shall disable network connectivity (i.e. Airplane Mode) and verify that the VPN disconnects. The evaluator shall reestablish network connectivity and verify that the VPN automatically reconnects.

Summary

Function not included in [ST] .

Assurance Activity AA-FMT_SMF_EXT.1-ATE-48

Function 46 [conditional]

The evaluator shall configure the TOE to use BAF and confirm that the biometric can be used to authenticate to the device. The evaluator shall revoke the biometric credential's ability to authenticate to the TOE and confirm that the same BAF cannot be used to authenticate to the device.

Summary

Function not included in [ST] .

Assurance Activity AA-FMT_SMF_EXT.1-ATE-49

Function 47

[The evaluator shall verify that the TSS describes all assigned security management functions and their intended behavior.]

- **Test 47:** The evaluator shall design and perform tests to demonstrate that the function may be configured and that the intended behavior of the function is enacted by the TOE.

Summary

Function not included in [ST] .

2.1.5.3 Extended: Specification of Specification of Remediation Actions (FMT_SMF_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FMT_SMF_EXT.2-ASE-01

The evaluator shall verify that the TSS describes all available remediation actions, when they are available for use, and any other administrator-configured triggers. The evaluator shall verify that the TSS describes how the remediation actions are provided to the administrator.

Summary

Section 8.5.4 in the [ST] describes the *Unenrollment*.

Section 8.5.4 references to [IOS_CFG], the Configuration Profile Key Reference, which describes the unenrollment options. The optional key PayloadRemovalDisallowed, if present and set to true, prevents the user from deleting the configuration profile, unless this profile has a password that the user can provide.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FMT_SMF_EXT.2-ATE-01

The evaluator shall use the test environment to iteratively configure the device to perform each remediation action in the selection. The evaluator shall configure the remediation action per how the TSS states it is provided to the administrator. The test environment could be a MDM agent application, but can also be an application with administrator access.

Summary

The evaluator wiped the device which removed the device from management. He then reinstalled the TOE as a managed device and performed a manual unenroll on the device and verified that the management profile was removed.

2.1.5.4 Trusted Policy Update (FMT_POL_EXT.2(AGENT))

FMT_POL_EXT.2.1

TSS Assurance Activities

Assurance Activity AA-FMT_POL_EXT.2.1-MDMA-ASE-01

The evaluator also ensures that the TSS (or the operational guidance) describes how the candidate policies are obtained by the MDM Agent; the processing associated with verifying the digital signature of the policy updates; and the actions that take place for successful (signature was verified) and unsuccessful (signature could not be verified) cases. The software components that are performing the processing must also be identified in the TSS and verified by the evaluators.

Summary

Section 8.5.2 in the [ST]  describes the *Configuration Profiles*. Section 8.10 in the [ST]  describes the *Mapping to the Security Functional Requirements*.

The evaluator ensured that the TSS describes that candidate policies obtained by the MDM agent can be digitally signed and encrypted, as described in paragraph 3 of section 8.5.2. The evaluator also ensured by looking at the entry FMT_POL_EXT.2 and verified that MDM policies can be signed against enterprise signatures.

Section 8.9.2.2 states that:

"Candidate policies are generated by the administrator and disseminated as a configuration profile using one of the methods already described in section 8.5.2 above."

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FMT_POL_EXT.2.1-MDMA-ATE-01

The evaluator shall perform a policy update from an available configuration interface (such as through a test MDM Server). The evaluator shall verify the update is signed and is provided to the MDM Agent. The evaluator shall verify the MDM Agent accepts the digitally signed policy.

The evaluator shall perform a policy update from an available configuration interface (such as through a test MDM Server). The evaluator shall provide an unsigned and an incorrectly signed policy to the MDM Agent. The evaluator shall verify the MDM Agent does not accept the digitally signed policy.

Summary

The evaluator deployed a new password policy to the device using the Apple Profile Manager and verified that it was successfully verified. He then created an invalid certificate with which to sign a profile and attempted to deploy this improperly signed profile and confirmed the device did not accept it, it asks the user whether or not to accept.

FMT_POL_EXT.2.2

TSS Assurance Activities

Assurance Activity AA-FMT_POL_EXT.2.2-MDMA-ASE-01

The assurance activity for this requirement is performed in conjunction with the assurance activity for FIA_X509_EXT.1 and FIA_X509_EXT.2.

Summary

Guidance Assurance Activities

Assurance Activity AA-FMT_POL_EXT.2.2-MDMA-AGD-01

The assurance activity for this requirement is performed in conjunction with the assurance activity for FIA_X509_EXT.1 and FIA_X509_EXT.2.

Summary

[CCGUIDE] [\[1\]](#) section 3.4.7.1 *Certificate Validation* references the "Certificate, Key, and Trust Services Reference" [CKTSREF] [\[1\]](#) as the API documentation related to certificate validation. This is covered by the function 'SetTrustEvaluate' whose usage is described in the section 'Trust' of [CKTSREF] [\[1\]](#).

Test Activities

Assurance Activity AA-FMT_POL_EXT.2.2-MDMA-ATE-01

The assurance activity for this requirement is performed in conjunction with the assurance activity for FIA_X509_EXT.1 and FIA_X509_EXT.2 as defined in the base PPs.

Summary

The evaluator deployed a password policy to the device using Apple Profile Manager and verified that when properly signed, it was accepted by the device, and when not properly signed or not signed at all, it was not accepted by the device.

2.1.5.5 Specification of Management Functions (FMT_SMF_EXT.3(AGENT))

FMT_SMF_EXT.3.1

TSS Assurance Activities

Assurance Activity AA-FMT_SMF_EXT.3.1-MDMA-ASE-01

The evaluator shall verify that the any assigned functions are described in the TSS and that these functions are documented as supported by the platform. The evaluator shall examine the TSS to verify that any differences between management functions and policies for each supported Mobile Device are listed.

Summary

Section 8.5.2 in the [ST] [\[1\]](#) describes the *Configuration Profiles*.

Since all the Mobile Devices specified in this [ST] [\[1\]](#) use iOS, there are no differences between supported management functions and policies between the different mobile devices. The supported management functions for iOS are described in [IOS_CFG] [\[1\]](#). The evaluator verified that the functions provided in FMT_SMF_EXT.3.1 are all configurable through the configuration profiles defined by the administrator (or MDM administrator) of the TOE.

Guidance Assurance Activities

Assurance Activity AA-FMT_SMF_EXT.3.1-MDMA-AGD-01

The evaluator shall verify the AGD guidance includes detailed instructions for configuring each function in this requirement.

If the MDM Agent is a component of the MDM system (i.e. MDM Server is the base PP), the evaluator shall verify, by consulting documentation for the claimed mobile device platforms, that the configurable functions listed for this Agent are supported by the platforms.

If the MDM Agent supports multiple interfaces for configuration (for example, both remote configuration and local configuration), the AGD guidance makes clear whether some functions are restricted to certain interfaces.

Summary

The evaluator noted that administrator-provided management functions in [PP_MD_V3.1] and [EP_MDM_AGENT_V3.0] are provided throughout [CCGUIDE]. A high-level summary is provided in Table 9 "Management Functions" of [CCGUIDE] where for each management function, a pointer to the provided guidance is provided. Also, these management functions are already assessed in prior work units of this report, in particular, the work units of AA-FMT_SMF_EXT.1-AGD.

Additionally, [CCGUIDE] provides Table 3 "SFR Configuration Requirements" outlining for each function (expressed in the form of an SFR and its description) whether configuration is needed and if needed which interface (e.g., API, Configuration Profile) can be used.

Test Activities

Assurance Activity AA-FMT_SMF_EXT.3.1-MDMA-ATE-01

Test 1: In conjunction with the assurance activities in the base PP, the evaluator shall attempt to configure each administrator-provided management function and shall verify that the Mobile Device executes the commands and enforces the policies.

Test 2: The evaluator shall configure the MDM Agent authentication certificate in accordance with the configuration guidance. The evaluator shall verify that the MDM Agent uses this certificate in performing the tests for FPT_ITT.1.

Test 3: (conditional) The evaluator shall design and perform tests to demonstrate that the assigned function may be configured and that the intended behavior of the function is enacted by the mobile device.

Summary

All tests from FMT_SMF_EXT.1.1 marked with an "X" in the column labeled "Administrator when enrolled in MDM" in [ST] table 5 were re-performed by the evaluator using the Profile Manager rather than the Apple Configurator 2 as was done for the tests in FMT_SMF_EXT.1.1.

FMT_SMF_EXT.3.2

TSS Assurance Activities

Assurance Activity AA-FMT_SMF_EXT.3.2-MDMA-ASE-01

The evaluator shall verify that the TSS describes the methods in which the MDM Agent can be enrolled.

The TSS description shall make clear if the MDM Agent supports multiple interfaces for enrollment and configuration (for example, both remote configuration and local configuration).

Additionally, the evaluator shall verify that the TSS describes any management functions of the MDM Agent, if assigned.

Summary

Section 8.4.3 in the [ST] describes the MDM server reference ID. Section 8.5.1 *Enrollment* in the [ST] describes how a device enrolls to an MDM server.

Section 8.5.1 describes that a device can enroll in multiple ways to an MDM server:

- Manually, using Apple's Profile Manager
- Manually, using Apple Configurator 2
- Distributing an enrollment profile via email, or a web site
- Device Enrollment Program (This is an automated and enforced method of automatically enrolling new devices.)

Section 8.4.3 also describes that iOS devices automatically connect to the MDM Server during setup if the device is enrolled into the Device Enrollment Program (DEP) and is assigned to an MDM Server. No other function was defined for the MDM agent than the pre-defined ones in FMT_SMF_EXT.3.2.

Guidance Assurance Activities

Assurance Activity AA-FMT_SMF_EXT.3.2-MDMA-AGD-01

The evaluator shall verify the AGD guidance includes detailed instructions for configuring each function in this requirement.

Summary

The evaluator noted that administrator-provided management functions in [PP_MD_V3.1] and [EP_MDM_AGENT_V3.0] are provided throughout [CCGUIDE]. A high-level summary is provided in Table 9 "Management Functions" of [CCGUIDE] where for each management function, a pointer to the provided guidance is provided. Additionally, these management functions are already assessed in prior work units of this report, in particular, the work units of AA-FMT_SMF_EXT.1-AGD.

Additionally, [CCGUIDE] provides Table 3 "SFR Configuration Requirements" outlining for each function (expressed in the form of an SFR and its description) whether configuration is needed and if needed which interface (e.g., API, Configuration Profile) can be used.

Test Activities

Assurance Activity AA-FMT_SMF_EXT.3.2-MDMA-ATE-01

Test 1: In conjunction with other assurance activities, the evaluator shall attempt to enroll the MDM Agent in management with each interface identified in the TSS, and verify that the MDM Agent can manage the device and communicate with the MDM Server.

Test 2: (conditional) In conjunction with the assurance activity for FAU_ALT_EXT.2.1, the evaluator shall configure the periodicity for reachability events for several configured time periods and shall verify that the MDM Server receives alerts on that schedule.

Test 3: (conditional) The evaluator shall design and perform tests to demonstrate that the assigned function may be configured and that the intended behavior of the function is enacted by the mobile device.

Summary

The evaluator used both manual enrollment and DEP enrollment to successfully create enrollment place holders for the TOE. Test devices are not fully enrolled, but the evaluator determined the process worked as required.

Only Test 1 is applicable to the TOE.

2.1.5.6 User Unenrollment Prevention (FMT_UNR_EXT.1(AGENT))

TSS Assurance Activities

Assurance Activity AA-FMT_UNR_EXT.1-MDMA-ASE-01

The evaluator shall ensure that the TSS describes the mechanism used to prevent users from unenrolling or the remediation actions applied when unenrolled.

Summary

Section 8.5.4 *Unenrollment* in the [ST] describes the unenrollment process for the TOE.

Section 8.5.4 describes that if the configuration profile key PayloadRemovalDisallowed, the user cannot delete the profile unless the profile has a removal password and the user provides it. It is up to the MDM server to set this key.

Guidance Assurance Activities

Assurance Activity AA-FMT_UNR_EXT.1-MDMA-AGD-01

The evaluator shall ensure that the administrative guidance instructs administrators in configuring the unenrollment prevention in each available configuration interface. If any configuration allows users to unenroll, the guidance also describes the actions that unenroll the Agent.

Summary

[CCGUIDE] [\[1\]](#) section 3.5.11 "TOE Unenrollment Prevention" provides information on how configure unenrollment prevention as follows:

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] [\[1\]](#), 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.

Test Activities

Assurance Activity AA-FMT_UNR_EXT.1-MDMA-ATE-01

If 'prevent the unenrollment from occurring' is selected:

Test 1: The evaluator shall configure the Agent according to the administrative guidance for each available configuration interface, shall attempt to unenroll the device, and shall verify that the attempt fails.

If 'apply remediation actions' is selected:

Test 2: If any configuration allows the user to unenroll, the evaluator shall configure the Agent to allow user unenrollment, attempt to unenroll, and verify that the remediation actions are applied.

Summary

For test 1, the evaluator set up the device according to the administrative guidance and found the "Remove" option was missing from the Device Management location in the UI, therefore the device cannot be unenrolled.

For test 2, the evaluator set the Profile Manager to allow unenrollment, set up the device according to the administrative guidance, and found that the "Remove" option was then available in the Device Management UI. He selected Remove and verified that the profile was removed, thus the device was unenrolled.

2.1.5.7 Specification of Management Functions (Wireless LAN) (FMT_SMF_EXT.1(WLAN))

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-FMT_SMF_EXT.1-WLAN-AGD-01

The evaluator shall check to make sure that every management function mandated by the EP is described in the operational guidance and that the description contains the information required to perform the management duties associated with the management function.

Summary

The evaluator noted that administrator-provided management functions defined in [PP_MD_V3.1] and [PP_WLAN_CLI_EP_V1.0] are provided throughout [CCGUIDE]. A high-level summary is provided in [CCGUIDE] Table 9 "Management Functions" where for each management function, a pointer to the provided guidance is provided. Also, these management functions are already assessed in prior work units of this report, in particular, the work units of AA-FMT_SMF_EXT.1-AGD.

Additionally, [CCGUIDE] provides Table 3 "SFR Configuration Requirements" outlining for each function (expressed in the form of an SFR and its description) whether configuration is needed and if needed which interface (e.g., API, Configuration Profile) can be used.

Test Activities

Assurance Activity AA-FMT_SMF_EXT.1-WLAN-ATE-01

The evaluator shall test the TOE's ability to provide the management functions by configuring the TOE and testing each option listed in the requirement above.

Note that the testing here may be accomplished in conjunction with the testing of other requirements, such as FCS_TLSC_EXT and FTA_WSE_EXT.

Summary

The evaluator individually tested each TLS cipher supported by the TOE using Safari to connect to a remote web site. He also tested connecting the TOE to a web server presenting both valid and invalid certificates and observed the expected behavior.

2.1.6 Protection of the TSF (FPT)

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

TSS Assurance Activities

Assurance Activity AA-FPT_AEX_EXT.1-ASE-01

The evaluator shall ensure that the TSS section of the ST describes how the 8 bits are generated and provides a justification as to why those bits are unpredictable.

Summary

Section 8.6.5 describes the *Domain Isolation*.

Paragraph 5 of section 8.6.5 describes that the 8 bits generated for ALSR are taken from the application processor TRNG involved in the randomization.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FPT_AEX_EXT.1-ATE-01

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

- **Test 1:** The evaluator shall select 3 apps included with the TSF. These must include any web browser or mail client included with the TSF. For each of these apps, the evaluator will launch the same app on two separate Mobile Devices of the same type and compare all memory mapping locations. The evaluator must ensure that no memory mappings are placed in the same location on both devices. If the rare (at most 1/256) chance occurs that two mappings are the same for a single app and not the same for the other two apps, the evaluator shall repeat the test with that app to verify that in the second test the mappings are different.

Summary

The evaluator performed this test on each of 3 common iOS applications, the Safari web browser, the Mail client, and the Messages application. Each application was run on two identical devices followed by a forced core dump. By synchronizing with iTunes, the evaluator could access the crash log and verify that the memory offsets for the same two applications were different on different platforms.

FPT_AEX_EXT.1.4

TSS Assurance Activities

Assurance Activity AA-FPT_AEX_EXT.1.4-ASE-01

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

Summary

This objective SFR element was not selected in the [ST] by the ST author.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

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

FPT_AEX_EXT.2.1

TSS Assurance Activities

Assurance Activity AA-FPT_AEX_EXT.2.1-ASE-01

The evaluator shall ensure that the TSS describes of the memory management unit (MMU), and ensures that this description documents the ability of the MMU to enforce read, write, and execute permissions on all pages of virtual memory.

Summary

Section 8.6.5 in the [ST]  describes the *Domain Isolation*.

Paragraph 6 in section 8.6.5 describes that the Memory Management Unit (MMU) supports memory address translation using a translation table maintained by the iOS kernel. The MMU maintains flags for every page of memory that allows/denies read, write or execute permissions on the data.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

FPT_AEX_EXT.2.2

TSS Assurance Activities

Assurance Activity AA-FPT_AEX_EXT.2.2-ASE-01

The evaluator shall ensure that the TSS describes of the memory management unit (MMU), and ensures that this description documents the ability of the MMU to enforce read, write, and execute permissions on all pages of virtual memory.

Summary

Section 8.6.5 in the [ST]  describes the *Domain Isolation*.

Section 8.6.5 describes the domain isolation for the applications in iOS. Every application is executed in its own domain (sandbox) which isolates the application from other applications. Applications are restricted to access other applications' files. The system files and resources are separated from the user's applications as well. The entire OS partition is mounted as read-only which prevents applications from modifying the system. Address Space Layout Randomization (ASLR) is also in place to protect against exploitation of memory corruption bugs. All applications are also signed which prevents an attacker from modifying an application on the device.

Section 8.6.5 specifies that

"the Memory Management Unit (MMU) supports memory address translation using a translation table maintained by the OS kernel. For each page, the MMU maintains flags that allow or deny the read, write or execution of data. Execution in this case allows the CPU to fetch instructions from a given page."

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.6.3 Extended: Anti-Exploitation Services (Overflow Protection) (FPT_AEX_EXT.3)**TSS Assurance Activities****Assurance Activity AA-FPT_AEX_EXT.3-ASE-01**

The evaluator shall determine that the TSS contains a description of stack-based buffer overflow protections implemented in the TSF software which runs in the non-privileged execution mode of the application processor. The exact implementation of stack-based buffer overflow protection will vary by platform. Example implementations may be activated through compiler options such as "-fstack-protector-all", "-fstackprotector", and "/GS" flags.

Summary

Section 8.10 describes the *Mapping to the Security Functional Requirements*.


Entry FPT_AEX_EXT.3 in the table in section 8.10 states that a

"Stack-based buffer overflow protection is implemented for every sandbox."

Assurance Activity AA-FPT_AEX_EXT.3-ASE-02

The evaluator shall ensure that the TSS contains an inventory of TSF binaries and libraries, indicating those that implement stack-based buffer overflow protections as well as those that do not. The TSS must provide a rationale for those binaries and libraries that are not protected in this manner.

Summary

Section 8.6.8 of the [ST]  states that a list of binaries has been provided to NIAP, since that list is considered proprietary information. However, all use space binaries are subject to ASLR.

Guidance Assurance Activities

No assurance activities defined.



Test Activities

No assurance activities defined.

2.1.6.4 Extended: Domain Isolation (FPT_AEX_EXT.4)**TSS Assurance Activities****Assurance Activity AA-FPT_AEX_EXT.4-ASE-01**

The evaluator shall ensure that the TSS describes the mechanisms that are in place that prevents non-TSF software from modifying the TSF software or TSF data that governs the behavior of the TSF. These mechanisms could range from hardware-based means (e.g. "execution rings" and memory management functionality); to software-based means (e.g. boundary checking of inputs to APIs). The evaluator determines that the described mechanisms appear reasonable to protect the TSF from modification.

Summary

Section 8.6.3 in the [ST]  describes the *Secure Software Update*. Section 8.6.5 in the [ST]  describes the *Domain Isolation*.

The entire OS partition is mounted as read-only which prevents any application or attacker to modify the system. System updates are signed by Apple, along with applications which are signed by their developers with their Apple developer certificate, which prevents any non-TSF software to modify the TSF software or TSF data.

Assurance Activity AA-FPT_AEX_EXT.4-ASE-02

The evaluator shall ensure the TSS describes how the TSF ensures that the address spaces of applications are kept separate from one another.

Summary

Section 8.6.5 in the [ST]  describes the *Domain Isolation*.

Section 8.6.5 describes the domain isolation for the apps in iOS. Every application is executed in its own domain (sandbox) which isolates the application from other applications. Applications are restricted from accessing other applications' files.

Assurance Activity AA-FPT_AEX_EXT.4-ASE-03

The evaluator shall ensure the TSS details the USSD and MMI codes available from the dialer at the locked state or during auxiliary boot modes that may alter the behavior of the TSF. The evaluator shall ensure that this description includes the code, the action performed by the TSF, and a justification that the actions performed do not modify user or TSF data. If no USSD or MMI codes are available, the evaluator shall ensure that the TSS provides a description of the method by which actions prescribed by these codes are prevented.

Summary

Section 8.6.5 in the [ST]  describes the *Domain Isolation*.

The TOE does not support Unstructured Supplementary Service Data (USSD) or Man-Machine Interface (MMI) codes and also does not support auxiliary boot modes.

Assurance Activity AA-FPT_AEX_EXT.4-ASE-04

The evaluator shall ensure the TSS documents any TSF data (including software, execution context, configuration information, and audit logs) which may be accessed and modified over a wired interface in auxiliary boot modes. The evaluator shall ensure that the description includes data, which is modified in support of update or restore of the device. The evaluator shall ensure that this documentation includes the auxiliary boot modes in which the data may be modified, the methods for entering the auxiliary boot modes, the location of the data, the manner in which data may be modified, the data format and packaging necessary to support modification, and software and/or hardware tools, if any, which are necessary for modifying the data.

Summary

Section 8.10 in the [ST]  describes the *Mapping to the Security Functional Requirements*.

The entry FPT_AEX_EXT.4 in the table in section 8.10 states that iOS does not support auxiliary boot modes. This assurance activity is therefore not applicable.

Assurance Activity AA-FPT_AEX_EXT.4-ASE-05

The evaluator shall ensure that the TSS provides a description of the means by which unauthorized and undetected modification (that is, excluding cryptographically verified updates per FPT_TUD_EXT.2) of the TSF data over the wired interface in auxiliary boots modes is prevented. The lack of publicly available tools is not sufficient justification. Examples of sufficient justification include auditing of changes, cryptographic verification in the form of a digital signature or hash, disabling the auxiliary boot modes, and access control mechanisms that prevent writing to files or flashing partitions.

Summary

Section 8.6.1 in the [ST] describes the *Secure Boot* process of iOS.

Section 8.6.1 describes that every step of the boot process contains components that are signed and which signature is verified during boot. This includes the bootloaders, kernel, kernel extensions and baseband firmware.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FPT_AEX_EXT.4-ATE-01

Assurance Activity Note: The following tests require the vendor to provide access to a test platform that provides the evaluator with tools that are typically not found on consumer Mobile Device products. In addition, the vendor provides a list of files (e.g., system files, libraries, configuration files, audit logs) that make up the TSF data. This list could be organized by folders/directories (e.g., /usr/sbin, /etc), as well as individual files that may exist outside of the identified directories.

- **Test 1:** The evaluator shall create and load an app onto the Mobile Device. This app shall attempt to traverse over all file systems and report any locations to which data can be written or overwritten. The evaluator must ensure that none of these locations are part of the OS software, device drivers, system and security configuration files, key material, or another untrusted application's image/data. For example, it is acceptable for a trusted photo editor app to have access to the data created by the camera app, but a calculator application shall not have access to the pictures.
- **Test 2:** For each available auxiliary boot mode, the evaluator shall attempt to modify a TSF file of their choosing using the software and/or hardware tools described in the TSS. The evaluator shall verify that the modification fails.

Summary

For test 1, the evaluator compiled and loaded an application to search all locations marked as writable. The evaluator verified all locations reported are within the application's home directory.

For test 2, the evaluator booted into iTunes mode and verified that no service is available to modify files.

2.1.6.5 Extended: JTAG Disablement (FPT_JTA_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FPT_JTA_EXT.1.1-ASE-01

If “disable access through hardware” is selected:

- The evaluator shall examine the TSS to determine the location of the JTAG ports on the TSF, to include the order of the ports (i.e., Data In, Data Out, Clock, etc.).

If “control access by a signing key” is selected:

- The evaluator shall examine the TSS to determine how access to the JTAG is controlled by a signing key.

Summary

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FPT_JTA_EXT.1.1-ATE-01

If “disable access through hardware” is selected the evaluator shall perform the following test:

- The evaluator shall connect a packet analyzer to the JTAG ports. The evaluator shall query the JTAG port for its device ID and confirm that the device ID cannot be retrieved.

If “control access by a signing key” is selected the evaluator shall perform the following test:

- The evaluator shall test with an application that is not approved to access the JTAG verifying that access cannot be achieved.

Summary

Section 8.6.2 of [ST] specifies the *Joint Test Action Group (JTAG) Disablement*. The evaluator found the TSS describes an interface that

"resembles the functionality of JTAG but does not implement the JTAG protocol."

The evaluator used a specialized test cable to connect to a specialized development device to access the JTAG-like interface. He then used the same specialized test cable connected to a regular TOE device and verified that the JTAG-like interface is not accessible.

2.1.6.6 Extended: Key Storage (FPT_KST_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FPT_KST_EXT.1-ASE-01

The evaluator shall consult the TSS section of the ST in performing the assurance activities for this requirement.

In performing their review, the evaluator shall determine that the TSS contains a description of the activities that happen on power-up and password authentication relating to the decryption of DEKs, stored keys, and data.

The evaluator shall ensure that the description also covers how the cryptographic functions in the FCS requirements are being used to perform the encryption functions, including how the KEKs, DEKs, and stored keys are unwrapped, saved, and used by the TOE so as to prevent plaintext from being written to non-volatile storage. The evaluator shall ensure that the TSS describes, for each power-down scenario how the TOE ensures that all keys in non-volatile storage are not stored in plaintext.

The evaluator shall ensure that the TSS describes how other functions available in the system (e.g., regeneration of the keys) ensure that no unencrypted key material is present in persistent storage.

The evaluator shall review the TSS to determine that it makes a case that key material is not written unencrypted to the persistent storage.

Summary

Section 8.2.1 in the [ST] describes the *Overview of Key Management* in iOS.

The evaluator verified that section 8.2.1 describes how the REK, KEKs and DEKs interact by hierarchically wrapping each other, and how these keys are generated or formed upon boot, or form the authentication of the user to iOS (by providing the correct passcode). Section 8.2.1 also describes which keys are wrapped, by which algorithm (namely AES in Key Wrap cipher mode), and which keys are stored in plaintext, encrypted, and in which type of memory (flash, non-volatile, etc.). The evaluator verified that the only key material that are written unencrypted in iOS are the REK (which is not accessible to any part of the system except the Secure Enclave), and the 0x89B and 0x835 keys. The latter are however stored in block 0 of the flash memory and can be erased very quickly if necessary. These keys are created during boot time and destroyed when the device is powered-off. When the device is locked, the passcode key is erased and can only be regenerated when the user provides the correct password.

Assurance Activity AA-FPT_KST_EXT.1-ASE-02

For each BAF selected in FIA_UAU.5.1:

The evaluator shall determine that the TSS also contains a description of the activities that happen on biometric authentication, relating to the decryption of DEKs, stored keys, and data. In addition how the system ensures that the biometric keying material is not stored unencrypted in persistent storage.

Summary

Section 8.4.1 describes the events during biometric authentication. Section 8.2.1 in the ST specifies the key management operations in iOS, and how KEK and other keys and derived and unwrapped based on the passcode used to unlock the device. The device passcode, and biometrics credentials are not stored on the device (according to section 8.4.2) - neither encrypted or unencrypted. The key derivation, combined with the salt and the UID, derives the passcode key used to unencrypt keys used to encrypt files on the system.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.6.7 Extended: No Key Transmission (FPT_KST_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FPT_KST_EXT.2-ASE-01

The evaluator shall determine that the TSS also contains a description of the activities that happen on biometric authentication, relating to the decryption of DEKs, stored keys, and data. In addition how the system ensures that the biometric keying material is not stored unencrypted in persistent storage.

In performing their review, the evaluator shall determine that the TSS contains a description of the activities that happen on power-up and password authentication relating to the decryption of DEKs, stored keys, and data.

The evaluator shall ensure that the TSS describes how other functions available in the system (e.g., regeneration of the keys) ensure that no unencrypted key material is transmitted outside the security boundary of the TOE.

The evaluator shall review the TSS to determine that it makes a case that key material is not transmitted outside the security boundary of the TOE.

Summary

Section 8.2.1 in the [ST] describes the *Overview of Key Management* in iOS.

The TOE is the mobile device running iOS. The cryptographic boundary is therefore the entire mobile device. The evaluator found in section 8.2.1 a description of the power-up process and password authentication within iOS relating to the generation, derivation, and decryption of DEKs, DEKs and stored data. Whenever a file is accessed, it gets decrypted by its respective class key. Then file is then decrypted with its respective file key. Whenever a file is closed, its file key is stored encrypted in memory. At any time any key is sent outside the security boundary of the TOE (i.e., outside the device).

Assurance Activity AA-FPT_KST_EXT.2-ASE-02

For each BAF selected in FIA_UAU.5.1:

n performing their review, the evaluator shall determine that the TSS contains a description of the activities that happen on biometric authentication, including how any plaintext material, including critical security parameters and results of biometric algorithms, are protected and accessed.

The evaluator shall ensure that the TSS describes how functions available in the biometric algorithms ensure that no unencrypted plaintext material, including critical security parameters and intermediate results, is transmitted outside the security boundary of the TOE or to other functions or systems that transmit information outside the security boundary of the TOE.

Summary

Section 8.4 of the ST describes the *Identification and Authentication (FIA)*. Section 8.5.2 in the ST describes the *Biometric Authentication Factors (BAFs)*.

The evaluator found that authentication credentials are not stored in the TOE in any form. When the user authenticates successfully, the key derivation output decrypts the keybag in the Secure Enclave Processor with the respective key class. No intermediate results are sent outside the TOE either; the key derivation process happens locally only.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.6.8 Extended: No Plaintext Key Export (FPT_KST_EXT.3)

TSS Assurance Activities

Assurance Activity AA-FPT_KST_EXT.3-ASE-01

The ST author will provide a statement of their policy for handling and protecting keys. The evaluator shall check to ensure the TSS describes a policy in line with not exporting either plaintext DEKs, KEKs, or keys stored in the secure key storage.

Summary

Section 8.1.1 in the [ST]  describes *The Secure Enclave*.

The evaluator verified that the second paragraph in section 8.1.1 contains a statement of the policy defining that

"no keys or key material may be exported."

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.



2.1.6.9 Extended: Self-Test Notification (FPT_NOT_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FPT_NOT_EXT.1-ASE-01

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

Summary

Section 8.6.1 in the [ST]  describes the *Secure Boot*. Section 8.6.9 in the [ST]  describes the *Self Tests*.

The evaluator reviewed sections 8.6.1 and 8.6.9 and verified that if the TOE encounters a failure of the self-tests the TOE will power itself off and will require a reboot. In case of a failure of the integrity check, the device will display the "Connect to iTunes" screen and transition to the recovery mode. In this mode, the device has to be plugged into a USB connection, connected to iTunes and factory reset.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

See assurance activities for SFR elements below.

FPT_NOT_EXT.1.1

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FPT_NOT_EXT.1.1-ATE-01

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

- **Test 1:** The evaluator shall use a tool provided by the developer to modify files and processes in the system that correspond to critical failures specified in the second list. The evaluator shall verify that creating these critical failures causes the device to take the remediation actions specified in the first list.

Summary

The FIPS 140-2 integrity and power-up test performed by the evaluator during the FIPS 140-2 validation satisfies this test. Previous agreement of NIAP and the validators determined this test does not need to be repeated.

2.1.6.10 Reliable Time Stamps (FPT_STM.1)

TSS Assurance Activities

Assurance Activity AA-FPT_STM.1-ASE-01

The evaluator shall examine the TSS to ensure that it lists each security function that makes use of time. The TSS provides a description of how the time is maintained and considered reliable in the context of each of the time related functions. This documentation must identify whether the TSF uses a NTP server or the carrier's network time as the primary time sources.

Summary

Section 8.6.7 in the [ST] describes the *Time*.

Section 8.6.7 defines which security functions in the TOE make usage of time:

- ALC_TSU_EXT
- FAU_GEN.1.2
- FIA_UAU.7
- FIA_X509_EXT.1.1
- FIA_X509_EXT.3.1
- FMT_SMF_EXT.1.1 Function 1
- FMT_SMF_EXT.1.1 Function 2
- FPT_STM.1.1
- FTA_SSL_EXT.1
- for A7 and later platforms: GPS or NTP (if GPS is not available)
- Network, Identity and Time Zone (NITZ)

Guidance Assurance Activities

Assurance Activity AA-FPT_STM.1-AGD-01

The evaluator examines the operational guidance to ensure it describes how to set the time.

Summary

[CCGUIDE][\[1\]](#) section 3.5.4 *Timestamp Configuration* provides instructions to set up the date and time which refers to [iPhone_UG][\[1\]](#) and [iPad_UG][\[1\]](#) *Get Started* section *Date and time* Also, this section states that in the evaluated configuration, the TOE must be configured to update its time automatically.

Test Activities

Assurance Activity AA-FPT_STM.1-ATE-01

- **Test 1:** *The evaluator uses the operational guide to set the time. The evaluator shall then use an available interface to observe that the time was set correctly.*

Summary

The evaluator disabled the automatic setting of time using the Network Time Protocol (NTP), changed the time, and verified that the updated time was displayed on the TOE screen.

2.1.6.11 Extended: TSF Cryptographic Functionality Testing (FPT_TST_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FPT_TST_EXT.1-ASE-01

The evaluator shall examine the TSS to ensure that it specifies the self-tests that are performed at start-up. This description must include an outline of the test procedures conducted by the TSF (e.g., rather than saying "memory is tested", a description similar to "memory is tested by writing a value to each memory location and reading it back to ensure it is identical to what was written" shall be used). The TSS must include any error states that they TSF may enter when self-tests fail, and the conditions and actions necessary to exit the error states and resume normal operation. The evaluator shall verify that the TSS indicates these self-tests are run at start-up automatically, and do not involve any inputs from or actions by the user or operator.

Summary

Section 8.6.9 in the [ST][\[1\]](#) describes the *Self-Tests*.

Section 8.6.9 describes which self-tests are run by iOS when the device is powered-up. The TOE performs the FIPS 140-2 power-on self-tests for its cryptographic algorithms, along with a software integrity test using HMAC-SHA-256. More information about the self-tests can be found in sections 8.6.9.1 through 8.6.9.2 in the [ST][\[1\]](#).

Assurance Activity AA-FPT_TST_EXT.1-ASE-02

The evaluator shall inspect the list of self-tests in the TSS and verify that it includes algorithm self-tests. The algorithm self-tests will typically be conducted using known answer tests.

Summary

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

2.1.6.12 Extended: TSF Integrity Testing (FPT_TST_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FPT_TST_EXT.2-ASE-01

The evaluator shall verify that the TSS section of the ST includes a description of the boot procedures, including a description of the entire bootchain, of the software for the TSF's Application Processor. The evaluator shall ensure that before loading the bootloader(s) for the operating system and the kernel, all bootloaders and the kernel software itself is cryptographically verified. For each additional category of executable code verified before execution, the evaluator shall verify that the description in the TSS describes how that software is cryptographically verified.

Summary

Section 8.6.1 in the [ST]  describes the *Secure Boot*.

Section 8.6.1 includes a description of the boot process, and how every component is signed and its signature is verified during boot time. This includes the kernel, kernel extensions, the bootloaders and baseband firmware.

Assurance Activity AA-FPT_TST_EXT.2-ASE-02

The evaluator shall verify that the TSS contains a justification for the protection of the cryptographic key or hash, preventing it from being modified by unverified or unauthenticated software. The evaluator shall verify that the TSS contains a description of the protection afforded to the mechanism performing the cryptographic verification.

Summary

Section 8.6.1 in the [ST]  describes the *Secure Boot*.

Section 8.6.1 describes that the baseband firmware's integrity is verified at boot by a digital signature algorithm. The asymmetric key used for this verification is protected in hardware.

Assurance Activity AA-FPT_TST_EXT.2-ASE-03

The evaluator shall verify that the TSS describes each auxiliary boot mode available on the TOE during the boot procedures. The evaluator shall verify that, for each auxiliary boot mode, a description of the cryptographic integrity of the executed code through the kernel is verified before each execution.

Summary

No auxiliary boot mode is claimed, and this assurance activity is therefore not applicable.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FPT_TST_EXT.2-ATE-01

The evaluator shall perform the following tests:

- **Test 1:** *The evaluator shall perform actions to cause TSF software to load and observe that the integrity mechanism does not flag any executables as containing integrity errors and that the TOE properly boots.*

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

- **Test 2:** *The evaluator shall modify a TSF executable that is integrity protected and cause that executable to be successfully loaded by the TSF. The evaluator observes that an integrity violation is triggered and the TOE does not boot. (Care must be taken so that the integrity violation is determined to be the cause of the failure to load the module, and not the fact that the module was modified so that it was rendered unable to run because its format was corrupt).*
- **Test 3:***[conditional] If the ST author indicates that the integrity verification is performed using a public key, the evaluator shall verify that the update mechanism includes a certificate validation according to FIA_X509_EXT.1. The evaluator shall digitally sign the TSF executable with a certificate that does not have the Code Signing purpose in the extendedKeyUsage field and verify that an integrity violation is triggered. The evaluator shall repeat the test using a certificate that contains the Code Signing purpose and verify that the integrity verification succeeds. Ideally, the two certificates should be identical except for the extendedKeyUsage field.*

Summary

For test 1, the evaluator successfully rebooted the TOE. Success of the boot process demonstrates no integrity errors occurred.

For test 2, the FIPS 140-2 integrity and power-up test was performed by the evaluator during the FIPS 140-2 validation, therefore it is not tested again.

Test 3 is not applicable to the TOE.

2.1.6.13 Extended: Trusted Update: TSF Version Query (FPT_TUD_EXT.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FPT_TUD_EXT.1-ATE-01

The evaluator shall establish a test environment consisting of the Mobile Device and any supporting software that demonstrates usage of the management functions. This can be test software from the developer, a reference implementation of management software from the developer, or other commercially available software. The evaluator shall set up the Mobile Device and the other software to exercise the management functions according to the provided guidance documentation.

- **Test 1:** *Using the AGD guidance provided, the evaluator shall test that the administrator and user can query:*
 - *the current version of the TSF operating system and any firmware that can be updated separately*
 - *the hardware model of the TSF*
 - *the current version of all installed mobile applications*

The evaluator must review manufacturer documentation to ensure that the hardware model identifier is sufficient to identify the hardware which comprises the device.

Summary

The evaluator was able to query the TOE for its current version of the operating system, its hardware model, and the current versions of all installed apps.

2.1.6.14 Extended: Trusted Update Verification (FPT_TUD_EXT.2)

TSS Assurance Activities

See assurance activities for SFR elements below.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FPT_TUD_EXT.2-ATE-01

The evaluator shall verify that the developer has provided evidence that the following tests were performed for each available update mechanism:

- **Test 1:** The tester shall try to install an update without the digital signature and shall verify that installation fails. The tester shall attempt to install an update with digital signature, and verify that installation succeeds.
- **Test 2:** The tester shall digitally sign the update with a key disallowed by the device and verify that installation fails. The tester shall digitally sign the update with the allowed key and verify that installation succeeds.
- **Test 3:** [conditional] The tester shall digitally sign the update with an invalid certificate and verify that update installation fails. The tester shall repeat the test using a valid certificate and a certificate that contains the purpose and verify that the update installation succeeds.
- **Test 4:** [conditional] The tester shall digitally sign the application with a certificate that does not have the Code Signing purpose and verify that application installation fails. The tester shall repeat the test using a valid certificate and a certificate that contains the Code Signing purpose and verify that the application installation succeeds.
- **Test 5:** [conditional] The tester shall repeat this test for the software executing on each processor listed in the first selection. The tester shall attempt to install an update without the digital signature and shall verify that installation fails. The tester shall attempt to install an update with digital signature, and verify that installation succeeds.

Summary

For test 1, the evaluator obtained a valid but unsigned image and attempted to load the image into the TOE. The image was rejected by the TOE.

For test 2, the evaluator signed the image with a disallowed key and attempted to load the image into the TOE. The image was rejected by the TOE. He also attempted to load a proper vendor-signed image and was successful.

Test 3 is not applicable to Apple.

For test 4, please see FPT_TUD_EXT.2.4.

Test 5 is not applicable to the TOE.

FPT_TUD_EXT.2.3

TSS Assurance Activities

Assurance Activity AA-FPT_TUD_EXT.2.3-ASE-01

The evaluator shall verify that the TSS section of the ST describes all TSF software update mechanisms for updating the system software. The evaluator shall verify that the description includes a digital signature verification of the software before installation and that installation fails if the verification fails. The evaluator shall verify that all software and firmware involved in updating the TSF is described and, if multiple stages and software are indicated, that the software/firmware responsible for each stage is indicated and that the stage(s) which perform signature verification of the update are identified.

Summary

Section 8.6.3 in the [ST]  describes the *Secure Software Updates*.

Section 8.6.3 describes the secure software updates provided by iOS. Whenever updates are available, users on iOS receive a notification that an update is available. All updates are digitally signed and users cannot downgrade to older software versions. These updates happen over a secure channel protected using the HTTPS protocol. The boot-time chain-of-trust makes sure the signature of the code comes from Apple. Paragraph 3 explains that the installation is only permitted if the verification process succeeds: the authorization server checks the presented list of measurements against versions for which installation is permitted. If the server finds a match, it adds the device unique ID to the measurement and signs the result. The server then passes a complete set of signed data to the device, which will then verify the signature, and install the upgrades if the signature verification is successful.

Assurance Activity AA-FPT_TUD_EXT.2.3-ASE-02

The evaluator shall verify that the TSS describes the method by which the digital signature is verified and that the public key used to verify the signature is either hardware-protected or is validated to chain to a public key in the Trust Anchor Database. If hardware-protection is selected, the evaluator shall verify that the method of hardware-protection is described and that the ST author has justified why the public key may not be modified by unauthorized parties.

Summary


Section 8.6.3 in the [ST]  describes the *Secure Software Updates*.

The software updates are digitally signed and verified using a hardware-protected asymmetric key used for signature verification. That asymmetric key is protected as a x509v3 certificate in the Secure Enclave (Apple's root certificate).

Assurance Activity AA-FPT_TUD_EXT.2.3-ASE-03

[conditional] If the ST author indicates that software updates to system software running on other processors is verified, the evaluator shall verify that these other processors are listed in the TSS and that the description includes the software update mechanism for these processors, if different than the update mechanism for the software executing on the Application Processor.

Summary

This selection is not included in the [ST]  by the ST author.

Assurance Activity AA-FPT_TUD_EXT.2.3-ASE-04

[conditional] If the ST author indicates that the public key is used for software update digital signature verification, the evaluator shall verify that the update mechanism includes a certificate validation according to FIA_X509_EXT.1 and a check for the Code Signing purpose in the extendedKeyUsage.

Summary

Section 8.2.1 in the [ST]  describes the *Overview of Key Management*.

Apple's root certificate is provided during manufacturing of the device, which is used to verify the authenticity of the software during boot and for updates of the system software. If the certificate is compromised, iOS can update certificates over wireless (this option can be disabled). The certificate validation and certificate path validation are implemented according to RFC5280 according to entry FIA_X509_EXT.1 of the table in section 8.10.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

No assurance activities defined.

FPT_TUD_EXT.2.4

TSS Assurance Activities

Assurance Activity AA-FPT_TUD_EXT.2.4-ASE-01

The evaluator shall verify that the TSS describes how mobile application software is verified at installation. The evaluator shall ensure that this method uses a digital signature.

Summary

Section 8.3 in the [ST] [\[1\]](#) describes the *User Data Protection (FDP)*.

Applications are signed and their signature is verified before installation by an Apple certificate. If the signature verification fails, the application is not installed.

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FPT_TUD_EXT.2.4-ATE-01

Assurance Activity Note: *The following test does not have to be tested using the commercial application store.*

- **Test 1:** *The evaluator shall write, or the developer shall provide access to, an application. The evaluator shall try to install this application without a digital signature and shall verify that installation fails. The evaluator shall attempt to install a digitally signed application, and verify that installation succeeds.*

Summary

The evaluator verified properly signed applications can be installed. Developer tools do not allow creation of unsigned applications. Test 3 is not applicable to the TOE since no additional certificates can be installed to authorize or limit apps.

2.1.6.15 TSF Cryptographic Functionality Testing (Wireless LAN) (FPT_TST_EXT.1(WLAN))



TSS Assurance Activities

Assurance Activity AA-FPT_TST_EXT.1-WLAN-ASE-01

The evaluator shall examine the TSS to ensure that it details the self tests that are run by the TSF on start-up; this description should include an outline of what the tests are actually doing (e.g., rather than saying "memory is tested", a description similar to "memory is tested by writing a value to each memory location and reading it back to ensure it is identical to what was written" shall be used). The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the TSF is operating correctly.

The evaluator shall examine the TSS to ensure that it describes how to verify the integrity of stored TSF executable code when it is loaded for execution. The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the integrity of stored TSF executable code has not been compromised. The evaluator also ensures that the TSS (or the operational guidance) describes the actions that take place for successful (e.g. hash verified) and unsuccessful (e.g., hash not verified) cases.

Summary

Section 8.6.9 *Self-Tests* in the [ST]  describes the self-tests performed by the TOE. Section 8.6.1 in the [ST]  describes the *Secure Boot* process.

Section 8.6.9 is split into several sections defining all the different self-tests implemented by the TOE. This includes power-up test, cryptographic algorithms tests (know-answer-tests or pair-wise consistency test), software/firmware integrity tests, and critical function tests. These self-tests are performed both for the CoreCrypto Kernel Module v8.0 for ARM (kernel space) and for the CoreCrypto Module v8.0 for ARM (user space). These sections clearly explain how the integrity of the libraries (libcorecrypto.dylib, libSystem.dylib and libcommoncrypto.dylib and the runtime image of the iOS kernel) is checked using the HMAC-SHA-256 algorithm. These sections also describe that if any of these tests should fail, the device shuts down automatically. Section 8.6.1 explains that if one step of the boot process, including the integrity verification of the stored TSF executable, fails, then the system does not boot and displays the "Connect to iTunes" screen, for device recovery. The evaluator verified that section 8.6.9.1 of the TSS provides an argument that the tests are sufficient:


" iOS ensures that there is only one physical instance of the library and maps it to all application linking to that library. In this way, the module always stays in memory. Therefore, the self-test during startup time is sufficient as it tests the module instance loaded in memory which is subsequently used by every application on iOS."

Guidance Assurance Activities

Assurance Activity AA-FPT_TST_EXT.1-WLAN-AGD-01

The evaluator shall ensure that the TSS (or the operational guidance) describes the actions that take place for successful (e.g. hash verified) and unsuccessful (e.g., hash not verified) cases.

Summary

Description of self-tests is provided in the TSS of [ST]  in section 8.6.9 *Self-Tests*. According to this section, self-tests are performed by the two Apple iOS CoreCrypto modules for ARM as outlined below.

Apple iOS CoreCrypto Module V8 for ARM

This crypto module performs self-tests to ensure the integrity of the module and the correctness of the cryptographic functionality at start up. In addition, the random bit generator requires continuous verification. The FIPS Self Tests application runs all required module self-tests. This application is invoked by the iOS startup process upon device power on.

The execution of an independent application for invoking the self-tests in the libcorecrypto.dylib makes use of features of the iOS architecture: the module, implemented in libcorecrypto.dylib, is linked by libcommoncrypto.dylib which is linked by libSystem.dylib. The libSystem.dylib is a library

that must be loaded into every application for operation. The library is stored in the kernel cache and therefore is not available on the disk as directly visible files. iOS ensures that there is only one physical instance of the library and maps it to all application linking to that library. In this way, the module always stays in memory. Therefore, the self-test during startup time is sufficient as it tests the module instance loaded in memory which is subsequently used by every application on iOS.

Self-tests performed by this module includes:

- Power-up tests: these tests (as listed in Table 18 of [ST]) are performed each time the Apple iOS CoreCrypto Module v8 starts and must be completed successfully for the module to operate in the FIPS approved mode. If any of the required tests fail, the device powers itself off. To rerun the self-tests on demand, the user must reboot the device.
- Software / Firmware integrity tests: A software integrity test is performed on the runtime image of the Apple iOS CoreCrypto Module v8 for ARM. The CoreCrypto's HMAC-SHA-256 is used as an approved algorithm for the integrity test. If the test fails, then the device powers itself off.
- Conditional tests covers the following tests:
 - Continuous Random Number Generator test is performed whenever CTR_DRBG is invoked.
 - Pair-wise consistency test: generate asymmetric keys and performs all required pair-wise consistency tests, the encryption/decryption as well as signature verification tests, with the newly generated key pairs.
 - SP 800-90A: the module performs a subset of the assurance tests as specified in section 11 of SP 800-90A, in particular it complies with the mandatory documentation requirements and performs known-answer tests and prediction resistance.

Apple iOS CoreCrypto Kernel Module V8 for ARM

This crypto module performs self-tests to ensure the integrity of the module and the correctness of the cryptographic functionality at start up. In addition, the DRBG requires continuous verification. The FIPS Self Tests functionality runs all required module self-tests. This functionality is invoked by the iOS Kernel startup process upon device initialization. If the self-tests succeed, the Apple iOS CoreCrypto Kernel Module V8.0 for ARM instance is maintained in the memory of the iOS Kernel on the device and made available to each calling kernel service without reloading.

Self-tests performed by this module includes:

- Power-up tests: these tests (as listed in Table 18 of [ST]) are performed each time the Apple iOS CoreCrypto Kernel Module V8.0 for ARM starts and must be completed successfully for the module to operate in the FIPS Approved Mode. If any of the following tests fails the device shuts down automatically. To run the self-tests on demand, the user may reboot the device.
- Software / Firmware integrity test: this test is performed on the runtime image of the Apple iOS CoreCrypto Kernel Module V8.0 for ARM. The CoreCrypto's HMAC-SHA256 is used as an approved algorithm for the integrity test. If the test fails, then the device powers itself off.
- Conditional tests covers the following tests:
 - Continuous Random Number Generator test is performed whenever CTR_DRBG is invoked. In addition, the seed source implemented in the operating system kernel also performs a continuous self-test.

- Pair-wise consistency test: the module generates asymmetric ECDSA key pairs and performs all required pair-wise consistency tests (signature generation and verification) with the newly generated key pairs.
- SP 800-90A: the module performs a subset of the assurance tests as specified in section 11 of SP800-90A, in particular it complies with the mandatory documentation requirements and performs know-answer tests and prediction resistance.

Test Activities

Assurance Activity AA-FPT_TST_EXT.1-WLAN-ATE-01

The evaluator shall perform the following tests:

- *Test 1: The evaluator performs the integrity check on a known good TSF executable and verifies that the check is successful.*
- *Test 2: The evaluator modifies the TSF executable, performs the integrity check on the modified TSF executable and verifies that the check fails.*

Summary

These tests were performed by the evaluator during the FIPS 140-2 validation. The test demonstrated that the integrity check of a known application succeeds and that of a changed application fails.

2.1.7 TOE access (FTA)

2.1.7.1 Extended: TSF- and User-initiated Locked State (FTA_SSL_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FTA_SSL_EXT.1-ASE-01

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

Summary

Section 8.6.6 in the [ST]  describes the *Device Locking*.

When the TOE transitions into a locked state, the class keys 'Complete Protection' and 'Accessible when unlocked' are wiped 10 seconds after the device is wiped, making all files in that class inaccessible.

Assurance Activity AA-FTA_SSL_EXT.1-ASE-02

The evaluator shall verify that the TSS describes the information allowed to be displayed to unauthorized users.

Summary

Section 8.7.3 of the [ST]  describes the *Lock Screen Banner Display*.



Unauthorized users are only capable of viewing the displayed banner on the device, to make emergency calls, use the flashlight and the camera.

Guidance Assurance Activities

Assurance Activity AA-FTA_SSL_EXT.1-AGD-01

The evaluation shall verify that the AGD guidance describes the method of setting the inactivity interval and of commanding a lock.

Summary

[CCGUIDE]  section 3.5.3 *Device/Session Locking* states that the TOE device is locked after a configurable time of user inactivity or upon request of the user. This can be defined by an administrator using a Configuration Profile by setting the configuration key "maxInactivity" in the Passcode Policy Payload to the desired time. [CCGUIDE]  section 3.4.1 *Passcode Authentication Configuration* provides a sample of such configuration profile.

User can set the time of user inactivity on their device by going to Settings>>Display &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.

Test Activities

Assurance Activity AA-FTA_SSL_EXT.1-ATE-01

- **Test 1:** *The evaluator shall configure the TSF to transition to the locked state after a time of inactivity (FMT_SMF_EXT.1) according to the AGD guidance. The evaluator shall wait until the TSF locks and verify that the display is cleared or overwritten and that the only actions allowed in the locked state are unlocking the session and those actions specified in FIA_UAU_EXT.2.*
- **Test 2:** *The evaluator shall command the TSF to transition to the locked state according to the AGD guidance as both the user and the administrator. The evaluator shall wait until the TSF locks and verify that the display is cleared or overwritten and that the only actions allowed in the locked state are unlocking the session and those actions specified in FIA_UAU_EXT.2.*

Summary

For test 1, the evaluator waited for the device to lock automatically and verified the screen was overwritten with non-sensitive data. He then confirmed that only allowed functions are available.

For test 2, the evaluator locked the device with the lock button and verified the screen was overwritten with non-sensitive data. He then confirmed that only allowed functions are available.

2.1.7.2 Default TOE Access Banners (FTA_TAB.1)

TSS Assurance Activities

Assurance Activity AA-FTA_TAB.1-ASE-01

The TSS shall describe when the banner is displayed.

Summary

Section 8.7.3 in the [ST]  describes the *Lock Screen Banner Display*.

A banner can be set up by the device and displayed when the device is locked (displayed on the lock screen).

Guidance Assurance Activities

No assurance activities defined.

Test Activities

Assurance Activity AA-FTA_TAB.1-ATE-01

The evaluator shall also perform the following test:

- **Test 1:** *The evaluator follows the operational guidance to configure a notice and consent warning message. The evaluator shall then start up or unlock the TSF. The evaluator shall verify that the notice and consent warning message is displayed in each instance described in the TSS.*

Summary

The banner is defined with a picture that is presented as part of the lock screen. The evaluator was able to set a new lock screen background and verify that it appeared when the screen was locked.

2.1.7.3 Wireless Network Access (FTA_WSE_EXT.1(WLAN))

TSS Assurance Activities

Assurance Activity AA-FTA_WSE_EXT.1-WLAN-ASE-01

The evaluator shall examine the TSS to determine that all of the attributes that can be used to specify acceptable networks (access points) are specifically defined.

Summary

Section 8.8.1 *EAP-TLS and TLS* in the [ST] [\[ST\]](#) describes the attributes that can be used to specify acceptable networks.

The evaluator examined section 8.8.1 and verified that the third to last paragraph in this section describes that when an application attempts to establish a trusted channel, the TOE will compare the DN contained within the peer certificate (CN, Subject Alternative Name fields, IP address or wild-cards) to the DN of the requested server. If the DN does not match the expected DN for the peer, the application cannot establish the connection.

Guidance Assurance Activities

Assurance Activity AA-FTA_WSE_EXT.1-WLAN-AGD-01

The evaluator shall examine the operational guidance to determine that it contains guidance for configuring each of the attributes identified in the TSS.

Summary

[CCGUIDE] [\[CCGUIDE\]](#) section 3.3.2 *VPN/Wi-Fi Configuration* describes VPN/Wi-Fi configuration. It states that 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] [\[IOS_CFG\]](#) and highlighted in [CCGUIDE] [\[CCGUIDE\]](#) section 3.1 *Configuration Profiles*. The evaluator examined [IOS_CFG] [\[IOS_CFG\]](#)

and determined that it provides the necessary information to configure the attributes specified in the TSS. Additionally, the evaluator examined the Table 4 in section 3.1 of [CCGUIDE] which describes the different attributes that can be specified in the configuration profile for a VPN payload.

Test Activities

Assurance Activity AA-FTA_WSE_EXT.1-WLAN-ATE-01

The evaluator shall also perform the following test for each attribute:

- *Test 1: The evaluator configures the TOE to allow a connection with a specific access point. The evaluator also configures the test environment such that the allowed access point and an access point that is not allowed are both "visible" to the TOE. The evaluator shall demonstrate that they can successfully establish a session with the allowed access point. The evaluator will then attempt to establish a session with the disallowed access point, and observe that the access attempt fails.*
- *Test 2: The evaluator configures the TOE to allow a connection with a specific access point using EAP-TLS authentication (not only will the valid SSID be configured but the TOE will also be provided with certificates to complete the EAP-TLS authentication). The evaluator also configures the test environment such that an access point broadcasts the SSID the TOE has been configured to connect to but the authentication server does not have valid credentials. The evaluator will then attempt to establish a session with the valid SSID/invalid authentication server, and observe that the access attempt fails.*

Summary

For test 1, the evaluator attempted to connect to both allowed and disallowed access points with the expected (success and failure) results.

For test 2, the evaluator has tested use of correct and incorrect authentication credentials while performing testing for FCS_TLSC_EXT.1(WLAN).

2.1.8 Trusted path/channels (FTP)

2.1.8.1 Trusted Channel Communication (FTP_ITC_EXT.1(2)(AGENT))

TSS Assurance Activities

Assurance Activity AA-FTP_ITC_EXT.1-MDMA-ASE-01

The evaluator shall examine the TSS to determine that the methods of Agent-Server communication are indicated, along with how those communications are protected. The evaluator shall also confirm that all protocols listed in the TSS in support of remote TOE administration are consistent with those specified in the requirement, and are included in the requirements in the ST.

Summary

Section 8.4.3 *MDM Server Reference ID* of the [ST] describes the MDM Server Reference ID. Section 8.8.1 *EAP-TLS and TLS* in the [ST] describes the EAP-TLS and TLS protocols and ciphersuites supported by the TOE.

Section 8.4.3 indicates that the agent and the server communicate using the HTTPS protocol (HTTP + TLS), using the ciphersuites indicated in section 8.8.1 for the TLS protocol. The evaluator verified that the protocols listed in the TSS (namely EAP-TLS and TLS supporting HTTPS) are consistent with the ones specified in the requirement and are included in the [ST].

Guidance Assurance Activities

Assurance Activity AA-FTP_ITC_EXT.1-MDMA-AGD-01

The evaluator shall confirm that the operational guidance contains instructions for configuring the communication channel between the MDM Agent and the MDM Server and conditionally, the MAS Server for each supported method.

Summary

[CCGUIDE] [\[1\]](#) section 3.2.12 *Configure MDM Agent and MDM Communications* provides information to configure the communications between the MDM Agent and MDM Server. It states as follows:

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* of [CCGUIDE] [\[1\]](#). 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, information is provided in section 3.2.9 *Certificate Authority (CA) Configuration* of [CCGUIDE] [\[1\]](#).

f

Test Activities

Assurance Activity AA-FTP_ITC_EXT.1-MDMA-ATE-01

For each supported identifier type (excluding DNs), the evaluator shall repeat the following tests:

Test 1: The evaluators shall ensure that communications using each specified (in the operational guidance) Agent-Server communication method is tested during the course of the evaluation, setting up the connections as described in the operational guidance and ensuring that communication is successful.

Test 2: The evaluator shall ensure, for each method of Agent-Server communication, the channel data is not sent in plaintext.

Test 3: The evaluator shall ensure, for each communication channel with the MDM Server, that a protocol analyzer identifies the traffic as the protocol under testing.

Further assurance activities are associated with the specific protocols.

Summary

The evaluator deployed a configuration change to the TOE using both the MDM and MAS communications methods. He also used Wireshark to confirm that the traffic was not sent in plaintext and that it was being sent with the intended communications protocol. These tests were performed during evaluation of FTP_ITC_EXT.1(WLAN).

2.1.8.2 Trusted Channel Communication (Wireless LAN) (FTP_ITC_EXT.1(WLAN))

TSS Assurance Activities

Assurance Activity AA-FTP_ITC_EXT.1-WLAN-ASE-01

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

Summary

Section 8.8.4 *Wireless LAN* in the [ST] [\[ST\]](#) describes the WLAN protocol implemented by the TOE. Section 8.8.1 *EAP-TLS and TLS* in the [ST] [\[ST\]](#) describes the EAP-TLS and TLS protocols and ciphersuites supported by the TOE.

These sections specify which protocols are implemented according to the IEEE 802.11 (2012) standard for WLAN and which algorithms are supported by the TOE. Section 8.8.1 specifies additional information about the EAP-TLS and TLS protocols certificates and ciphersuites that can be requested by applications.

Guidance Assurance Activities

Assurance Activity AA-FTP_ITC_EXT.1-WLAN-AGD-01

The evaluator shall confirm that the operational guidance contains instructions for establishing the connection to the access point, and that it contains recovery instructions should a connection be unintentionally broken.

Summary

[CCGUIDE] [\[CCGUIDE\]](#) section 3.4.7.1 *Certificate Validation* states that 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.

Additionally, configuration for EAP-TLS is provided in [CCGUIDE] [\[CCGUIDE\]](#) section 3.2.7 *EAP-TLS Configuration*. It states that WPA-EAP should be configured as follows:

“

- 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

”

Additionally, no recovery instructions was provided for unintentional broken connection as iOS will always attempt reconnection on its own to a trusted access point in range.

Test Activities

Assurance Activity AA-FTP_ITC_EXT.1-WLAN-ATE-01

The evaluator shall perform the following tests:

- *Test 1: The evaluators shall ensure that the TOE is able to initiate communications with an access point using the protocols specified in the requirement, setting up the connections as described in the operational guidance and ensuring that communication is successful.*
- *Test 2: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data is not sent in plaintext.*
- *Test 3: The evaluator shall ensure, for each communication channel with an authorized IT entity, modification of the channel data is detected by the TOE.*
- *Test 4: The evaluators shall physically interrupt the connection from the TOE to the access point (e.g., moving the TOE host out of range of the access point, turning the access point off). The evaluators shall ensure that subsequent communications are appropriately protected, at a minimum in the case of any attempts to automatically resume the connection or connect to a new access point.*

Further assurance activities are associated with the specific protocols.

Summary

For tests 1-2, the evaluator confirmed that the communication establishment is successful. He then used Wireshark to confirm that the traffic is encrypted.

For test 3, the evaluator verified that the protocol used is TLS.

For test 4, the evaluator established a connection and then shut down the access point. After verifying the connection is broken, the evaluator restarted the access point and verified that the TOE automatically re-established the connection to the access point.

2.2 Security Assurance Requirements

2.2.1 Life-cycle support (ALC)

2.2.1.1 Labelling of the TOE (ALC_CMC.1)

Assurance Activity AA-ALC_CMC.1-ALC-01

The evaluator shall check the ST to ensure that it contains an identifier (such as a product name/version number) that specifically identifies the version that meets the requirements of the ST. Further, the evaluator shall check the AGD guidance and TOE samples received for testing to ensure that the version number is consistent with that in the ST. If the vendor maintains a web site advertising the TOE, the evaluator shall examine the information on the web site to ensure that the information in the ST is sufficient to distinguish the product.

Summary

[ST] section 1.2 *TOE Reference* identifies the TOE as "Apple iPad and iPhone Mobile Devices with iOS 11.2" specified in section 1.4 *TOE Description* of [ST]. These devices use either A7 processor (iPhone 5s, iPad mini 3), A8/A8X processor (iPhone 6, iPhone 6 Plus, iPad mini 4, iPad Air 2), A9/A9X processor (iPhone 6s Plus, iPhone 6s, iPhone SE, iPad Pro 9.7", iPad Pro 12.9", iPad), A10 Fusion/A10X Fusion processor (iPhone 7 Plus, iPhone 7, iPad Pro 12.9", iPad Pro 10.5"), and A11 Bionic processor (iPhone 8, iPhone 8 Plus, iPhone X).

The guidance documentation, for example, [CCGUIDE] contains TOE reference as iOS 11.2.

In addition, for the delivery of the evaluated TOE devices, [CCGUIDE] section 2.1 *Secure Installation and Delivery of the TOE* states the following:

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

The normal distribution channels for a regular end user to obtain these devices include:

- The Apple Store (either a physical stores or online at <http://www.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" (<http://www.apple.com>), go to the upper left of the page and click "Business Store Home." Or optionally, use the following link.

- http://www.apple.com/us_smb_78313/shop

Government

Government customers can use the following link.

- <http://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.

The developer maintains a website (<https://support.apple.com/en-us/HT202739>) advertising the TOE. The evaluator examined the website and found that it identifies the TOE as generically as iOS 11 as being currently "In Evaluation" conforming to MDFPP3.1 ([PP_MD_V3.1] [\[1\]](#)), MDMAgentEP3.0 ([EP_MDM_AGENT_V3.0] [\[1\]](#)), and (WLAN Agent) ([PP_WLAN_CLI_EP_V1.0] [\[1\]](#)).

2.2.1.2 TOE CM coverage (ALC_CMS.1)

Assurance Activity AA-ALC_CMS.1-ALC-01

The evaluator shall ensure that the developer has identified (in public-facing development guidance for their platform) one or more development environments appropriate for use in developing applications for the developer's platform. For each of these development environments, the developer shall provide information on how to configure the environment to ensure that buffer overflow protection mechanisms in the environment(s) are invoked (e.g., compiler and linker flags). The evaluator shall ensure that this documentation also includes an indication of whether such protections are on by default, or have to be specifically enabled.

Summary

The evaluator examined the provided evidence which contains the following information:

The developer uses the Xcode IDE (Integrated Development Environment) is used by developers for building apps for Mac, iPhone, and iPad. Xcode IDE is tightly integrated with the Cocoa and Cocoa Touch frameworks of the iOS. Xcode comes with an "Assistant editor" that provides intuitive display of the source code. Additionally, Apple LLVM compiler parses the code, keeping every symbol in the LLBD debugger consistent with the editor and compiler and finding mistakes and offering Fix-its for the code.

The vendor provided "Secure Coding Guide" [SecCodeGuide] [\[1\]](#) which provides in section *Avoiding Buffer Overflows and Underflows* advice on how to avoid buffer overflows and underflows. In particular, subsection *Security Features that Can Help* states that iOS provides two features that can make it harder to exploit stack and buffer overflows which are address space layout randomization (ASLR) and non-executable stack and heap which are described as follows.

Address Space Layout Randomization

Recent versions of OS X and iOS, where possible, choose different locations for your stack, heap, libraries, frameworks, and executable code each time you run your software. This makes it much harder to successfully exploit buffer overflows because it is no longer possible to know where the buffer is in memory, nor is it possible to know where libraries and other code are located

Address space layout randomization requires some help from the compiler—specifically, it requires position-independent code.

- *If you are compiling an executable that targets OS X v10.7 and later (-macosx_version_min) or iOS v4.3 and later (-ios_version_min), the necessary flags are enabled by default. You can disable this feature, if necessary, with the -no_pie flag, but for maximum security, you should not do so.*
- *If you are compiling an executable that targets an earlier OS, you must explicitly enable position-independent executable support by adding the -pie flag.*

Non-Executable Stack and Heap

Recent processors support a feature called the NX bit that allows the operating system to mark certain parts of memory as non-executable. If the processor tries to execute code in any memory page marked as non-executable, the program in question crashes.

OS X and iOS take advantage of this feature by marking the stack and heap as non-executable. This makes buffer overflow attacks harder because any attack that places executable code on the stack or heap and then tries to run that code will fail.

Most of the time, this is the behavior that you want. However, in some rare situations (such as writing a just-in-time compiler), it may be necessary to modify that behavior.

There are two ways to make the stack and heap executable:

- Pass the `-allow_stack_execute` flag to the compiler. This makes the stack (not the heap) executable.
- Use the `mprotect` system call to mark specific memory pages as executable.

Assurance Activity AA-ALC_CMS.1-ALC-02

The evaluator shall ensure that the TSF is uniquely identified (with respect to other products from the TSF vendor), and that documentation provided by the developer in association with the requirements in the ST is associated with the TSF using this unique identification.

Summary

According to [ST], the TOE is Apple iPad and iPhone with iOS 11.2 and claims exact conformance to [PP_MD_V3.1], [EP_MDM_AGENT_V3.0], and [PP_WLAN_CLI_EP_V1.0]. The exact conformance requires the TOE to be capable of providing/supporting the TSF defined in the aforementioned PP and EP's. As assessed in the work units of ALC_CMS.1E, the evaluator found that the provided TOE documentation associated with the TOE contains unique identification which is generically iOS 11 (e.g., [iPad_UG]) or precisely iOS 11.2 (e.g., [CCGUIDE]).

2.2.1.3 Extension: Timely Security Updates (ALC_TSU_EXT.1)

Assurance Activity AA-ALC_TSU_EXT.1-ALC-01

The evaluator shall verify that the TSS contains a description of the timely security update process used by the developer to create and deploy security updates. The evaluator shall verify that this description addresses the TOE OS, the firmware, and bundled applications, each. The evaluator shall also verify that, in addition to the TOE developer's process, any carrier or other third-party processes are also addressed in the description. The evaluator shall also verify that each mechanism for deployment of security updates is described.

The evaluator shall verify that, for each deployment mechanism described for the update process, the TSS lists a time between public disclosure of a vulnerability and public availability of the security update to the TOE patching this vulnerability, to include any third-party or carrier delays in deployment. The evaluator shall verify that this time is expressed in a number or range of days.

The evaluator shall verify that this description includes the publicly available mechanisms (including either an email address or website) for reporting security issues related to the TOE. The evaluator shall verify that the description of this mechanism includes a method for protecting the report either using a public key for encrypting email or a trusted channel for a website.

The evaluator shall verify that the description includes where users can seek information about the availability of new security updates including details of the specific public vulnerabilities corrected by each update. The evaluator shall verify that the description includes the minimum amount of time that the TOE is expected to be supported with security updates, and the process by which users can seek information about when the TOE is no longer expected to receive security updates.

Summary

The Security Target [ST] section 8.6.3 *Secure Software Updates* describes the secure software update process for the TOE as follows:

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, and updates are delivered wirelessly, encouraging rapid adoption of the latest security fixes.

The startup process described above 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's been fixed in the newer version.

On a device with an A7 or later A-series processor, the Secure Enclave coprocessor 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 via HTTPS trusted channel. With iTunes, a full copy of iOS is downloaded and installed. OTA software updates download only the components required to complete an update, improving network efficiency, rather than downloading the entire OS. 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, LLB, iBoot, the kernel, and OS image), a random anti-replay value (nonce), and the device's unique 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 steps ensure that the authorization is for a specific device and that an old iOS version from one device can't 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.

The developer describes the delivery time of the TOE updates as follows:

The delivery time for updates is based on several factors including the time taken to receive, confirm and determine the severity of the issue; the time taken to resolve the issue, including development and testing of the solution; the time taken to deploy any resulting patches or guidance. This process can vary in length from two days to several months. All updates are then immediately made available worldwide to all devices directly from Apple and only Apple including those platform updates that directly support third-party infrastructure changes such as a carrier update.

Apple provides several forms of communication and guidance relating to product security. A public facing security web page *Contact Apple About Security Issues* at <https://support.apple.com/en-us/HT201220> provides clarification and links for assistance to

customers, security and privacy researchers, and law enforcement. Apple account (Apple ID) passwords, privacy and security tips, technical support for keeping software up-to-date, and how to join the security-announce mailing list are just a few of the helpful links highlighted here.

Once a security update is available, it is added to the posted list for all platforms on the web page *Apple security updates* at <https://support.apple.com/en-us/HT1222>. Here you will find detailed descriptions of potential security vulnerabilities with their corresponding Common Vulnerabilities and Exposures (CVE) identifier which are searchable at <https://nvd.nist.gov>.

For references to key Apple product certifications, validations and global government security guidance, Apple provides a comprehensive web page *Product security certifications, validations, and guidance for iOS* maintaining the relevant descriptions and links to those resources at <https://support.apple.com/en-us/HT202739>.

The sponsor Apple, Inc. stated to the evaluator that "Apple has historically never provided a statement on the length of time between public disclosure of a vulnerability and the public availability of security updates to the TOE. The total time has varied between one day to one week depending on the severity and complexity of a vulnerability."

The evaluator examined the description above and the determined the following:

- The description of the timely security update process used by the developer to create and deploy security updates is consistent with the description in [ST] section 8.6.3 *Secure Software Update*.
- The description addresses the TOE operating system.
- The description does not include any carrier or other third-party processes.
- The description covers the supported mechanism (described above) for deployment of security updates.
- The description includes the publicly available mechanism (e.g., Apple support site <https://www.apple.com/support/security/>) for providing security issues related to the TOE. Also, this support site is protected via HTTPS.
- The description includes where users can seek information about the availability of new security updates including details of specific public vulnerabilities corrected by each update. Additional information can be found on the support website <https://support.apple.com/en-us/HT201222>.
- The description provides the link in the previous item which contains information about the minimum amount of time that the TOE is expected to be supported with security updates, and the process by which users can seek information about when the TOE is no longer expected to receive security updates.

The evaluator determined that the description contains the necessary information.

2.2.2 Security Target evaluation (ASE)

2.2.3 Guidance documents (AGD)

2.2.3.1 Operational user guidance (AGD_OPE.1)

Assurance Activity AA-AGD_OPE.1-AGD-01

Some of the contents of the operational guidance are verified by the assurance activities in Section 5.1 and evaluation of the TOE according to the [CEM]. The following additional information is also required.

The operational guidance shall contain a list of natively installed applications and any relevant version numbers. If any third party vendors are permitted to install applications before purchase by the end user or enterprise, these applications shall also be listed.

The operational guidance shall contain instructions for configuring the cryptographic engine associated with the evaluated configuration of the TOE. It shall provide a warning to the administrator that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.


The documentation must describe the process for verifying updates to the TOE by verifying a digital signature. The evaluator shall verify that this process includes the following steps:

- *Instructions for obtaining the update itself. This should include instructions for making the update accessible to the TOE (e.g., placement in a specific directory).*
- *Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes generation of the hash/digital signature.*


The TOE will likely contain security functionality that does not fall in the scope of evaluation under this PP. The operational guidance shall make it clear to an administrator which security functionality is covered by the evaluation activities.

Summary

Installed apps

[CCGUIDE]  section 3.8 *Installed Apps* provides Table 12 listing the natively installed and free apps on the TOE iPad and iPhone devices.

Cryptographic modules


[CCGUIDE]  section 3.2 *Crypto-Related Function Configuration* provides information about the cryptographic modules that provide cryptographic support to the TOE. It states that the TOE comes with the following two cryptographic modules that provide the cryptographic support for the TOE:

- Apple iOS CoreCrypto Kernel Module v8 for ARM
- Apple iOS CoreCrypto Module v8 for ARM

Section 3.2 also contains the following warning:

"Warning: *Use of any other cryptographic module that was not evaluated or testing during Common Criteria of the TOE will take the operating environment out of the evaluated configuration."*

Secure software updates

[CCGUIDE]  section 2.2 *Secure Software Updates* describes the procedure for obtaining and verifying updates to the TOE. The process is as follows:

"

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.

"

Security functionality provided by the TOE:

[CCGUIDE]  section 1.4 *TOE Security Functionality (TSF)* lists at a high-level the security functionality provided by the TOE in the evaluated configuration which covers:

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

While assessing the assurance activities for the guidance documentation, the evaluator also verified that the guidance covers the security functionalities listed above. The evaluator noticed that in particular, [CCGUIDE]  chapter 3 *Administrative Guidance* is largely structured based on these security functionalities, for example, section 3.4 and section 3.5 are labeled as *Identification & Authorization Configuration* and *Security Management Configuration*, respectively. This structure

makes it easy for the reader to follow as well as for the evaluator to verify which security functionalities from [PP_MD_V3.1][\[1\]](#), [EP_MDM_AGENT_V3.0][\[1\]](#), and [PP_WLAN_CLI_EP_V1.0][\[1\]](#) are covered in the guidance/assurance activities.

Furthermore, section 2.3 *Unevaluated Functionalities* of [CCGUIDE][\[1\]](#) explicitly lists the security functionalities that are outside the scope of the evaluated configuration.

2.2.3.2 Preparative procedures (AGD_PRE.1)

Assurance Activity AA-AGD_PRE.1-AGD-01

As indicated in the introduction of section 5.2.3, there are significant expectations with respect to the documentation—especially when configuring the operational environment to support TOE functional requirements. The evaluator shall check to ensure that the guidance provided for the TOE adequately addresses all platforms claimed for the TOE in the ST.

Summary

While performing other work units the evaluator determined that sufficient guidance was provided for the TOE to address all claimed platforms. The guidance provided for the TOE applies to all platforms claimed in [ST][\[1\]](#) for the TOE.

2.2.4 Tests (ATE)

2.2.4.1 Independent testing - conformance (ATE_IND.1)

Assurance Activity AA-ATE_IND.1-ATE-01

The evaluator shall prepare a test plan and report documenting the testing aspects of the system. The test plan covers all of the testing actions contained in the [CEM] and the body of this PP's Assurance Activities. While it is not necessary to have one test case per test listed in an Assurance Activity, the evaluator must document in the test plan that each applicable testing requirement in the ST is covered.

The test plan identifies the platforms to be tested, and for those platforms not included in the test plan but included in the ST, the test plan provides a justification for not testing the platforms. This justification must address the differences between the tested platforms and the untested platforms, and make an argument that the differences do not affect the testing to be performed. It is not sufficient to merely assert that the differences have no affect; rationale must be provided. If all platforms claimed in the ST are tested, then no rationale is necessary.

The test plan describes the composition of each platform to be tested, and any setup that is necessary beyond what is contained in the AGD documentation. It should be noted that the evaluator is expected to follow the AGD documentation for installation and setup of each platform either as part of a test or as a standard pre-test condition. This may include special test drivers or tools. For each driver or tool, an argument (not just an assertion) should be provided that the driver or tool will not adversely affect the performance of the functionality by the TOE and its platform. This also includes the configuration of the cryptographic engine to be used. The cryptographic algorithms implemented by this engine are those specified by this PP and used by the cryptographic protocols being evaluated (IPsec, TLS/HTTPS, SSH).

The test plan identifies high-level test objectives as well as the test procedures to be followed to achieve those objectives. These procedures include expected results. The test report (which could just be an annotated version of the test plan) details the activities that took place when the test procedures were executed, and includes the actual results of the tests. This shall be a cumulative account, so if there was a test run that resulted in a failure; a fix installed; and then a successful re-run of the test, the report would show a "fail" and "pass" result (and the supporting details), and not just the "pass" result.

Summary

The evaluator prepared and executed the test plan [DTR]¹ to cover the Assurance Activities required by [PP_MD_V3.1]¹, [EP_MDM_AGENT_V3.0]¹, and [PP_WLAN_CLI_EP_V1.0]¹. This test plan identifies the TOE platforms, test prerequisites, steps for each test, expected results, and actual observed results.

2.2.5 Vulnerability assessment (AVA)

2.2.5.1 Vulnerability survey (AVA_VAN.1)

Assurance Activity AA-AVA_VAN.1-AVA-01

The evaluator shall generate a report to document their findings with respect to this requirement. This report could physically be part of the overall test report mentioned in ATE_IND, or a separate document. The evaluator performs a search of public information to find vulnerabilities that have been found in mobile devices and the implemented communication protocols in general, as well as those that pertain to the particular TOE. The evaluator documents the sources consulted and the vulnerabilities found in the report

For each vulnerability found, the evaluator either provides a rationale with respect to its non-applicability, or the evaluator formulates a test (using the guidelines provided in ATE_IND) to confirm the vulnerability, if suitable. Suitability is determined by assessing the attack vector needed to take advantage of the vulnerability. If exploiting the vulnerability requires expert skills and an electron microscope, for instance, then a test would not be suitable and an appropriate justification would be formulated.

Summary

The evaluator searched ¹ for publicly available vulnerability reports for iOS from the following sources:

- Apple security content disclosure statements for releases of iOS 11 related to this evaluation:
 - <https://support.apple.com/en-us/HT208143>, iOS 11.0.1, September 26, 2017
 - <https://support.apple.com/en-us/HT208164>, iOS 11.0.2, October 3, 2017
 - <https://support.apple.com/en-us/HT208182>, iOS 11.0.3, October 11, 2017
 - <https://support.apple.com/en-us/HT208222>, iOS 11.1, October 31, 2017
 - <https://support.apple.com/en-us/HT208255>, iOS 11.1.1, November 9, 2017
 - <https://support.apple.com/en-us/HT208282>, iOS 11.1.2, November 16, 2017
 - <https://support.apple.com/en-us/HT208334>, iOS 11.2, December 2, 2017
 - <https://support.apple.com/en-us/HT208357>, iOS 11.2.1, December 13, 2017
 - <https://support.apple.com/en-us/HT208401>, iOS 11.2.2, January 8, 2018
 - <https://support.apple.com/en-us/HT208463>, iOS 11.2.5, January 23, 2018
 - <https://support.apple.com/en-us/HT208534>, iOS 11.2.6, February 19, 2018
- MITRE Common Vulnerabilities and Exposures (CVE) List:
 - <https://cve.mitre.org/cve/cve.html>
- NIST National Vulnerability Database (NVD):
 - <https://web.nvd.nist.gov/view/vuln/search>

iOS provided no releases numbered 11.2.3 or 11.2.4.

The following search terms were used on the MITRE and NIST web sites:

- ios ipad
- ios iphone

¹ Latest search performed on 2018-03-26.

- ios core tls
- ios core crypto
- ios common crypto
- ios http
- ios https
- ios tcp
- ios ip
- ios bluetooth
- ios ipsec
- ios mdm
- ios mobile
- broadcom wifi crypto

A Appendixes

A.1 References

AConfig	Apple Configurator 2 Help (online guidance) Date received 2017-03-20 Location http://help.apple.com/configurator/mac/2.2/
APIGuide	App Programming Guide for iOS Date 2017-03-01
CC	Common Criteria for Information Technology Security Evaluation Version 3.1R5 Date April 2017 Location http://www.commoncriteriaportal.org/files/ccfiles/CCPART1V3.1R5.pdf Location http://www.commoncriteriaportal.org/files/ccfiles/CCPART2V3.1R5.pdf Location http://www.commoncriteriaportal.org/files/ccfiles/CCPART3V3.1R5.pdf
CCEVS-TD0194	Update to Audit of FTP_ITC_EXT.1/WLAN Date 2017-04-11 Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?td_id=198
CCEVS-TD0236	FCS_TLSC_EXT.2.1 - TLS Client Curve Limitation Date 2017-09-08 Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?td_id=242
CCEVS-TD0237	FAU_GEN.1.1(2) - FMT_UNR_EXT.1 Audit Record Selection-Based Date 2017-09-26 Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?td_id=243
CCEVS-TD0244	FCS_TLSC_EXT - TLS Client Curves Allowed Date 2017-11-16 Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?td_id=250
CCGUIDE	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 Author(s) atsec Version 1.01 Date 2018-03-30 File name agd/st-vid10851_agd.pdf
CC-MAN	Common Crypto man pages Date received 2018-02-02 Location https://developer.apple.com/legacy/library/documentation/Darwin/Reference/ManPages/#

CEM	Common Methodology for Information Technology Security Evaluation Version 3.1R5 Date April 2017 Location http://www.commoncriteriaportal.org/files/ccfiles/CEMV3.1R5.pdf
CKTSREF	Certificate, Key, and Trust Services (API Reference) Date received 2017-03-31 Location https://developer.apple.com/library/mac/documentation/Security/Reference/certifkeytrustservices/index.html
CRYPTOGUIDE	Cryptographic Services Guide Date 2014-07-15 Location https://developer.apple.com/library/ios/documentation/Security/Conceptual/cryptoservices/Introduction/Introduction.html File name agd/cryptoservices.pdf
DTR	Test Plan and Detailed Test Report - Apple iOS 11.0 VID10851 Version 2.1 Date 2018-03-28 File name ind/VID10851_TestPlan_TestResults_v2.1.pdf
EAR	Apple iOS 11 Entropy Assessment Report Version 4.3 Date received 2017-11-23 File name ase/VID10851_EAR_Apple_iOS11-v4.3.pdf
EP_MDM_AGENT_V3.0	Extended Package for Mobile Device Management Agents Version 3.0 Version 3.0 Date 2016-11-21 Location https://www.niap-ccevs.org/pp/ep_mdm_agent_v3.0.pdf File name ase/ep_mdm_agent_v3.0.pdf
FSPMapping	Apple iOS 10.2 FSP Mapping Date 2017-06-14 File name adv/fsp/Apple_iOS_10_FSP.xlsx
HTTPSTN2232	Keychain Services Programming Guide Date 2014-07-29 Location https://developer.apple.com/library/ios/technotes/tn2232/_index.html
HWSPEC	Hardware spec Date December 2017 File name alc/DeviceInformation-2017.12-iOS11-v7.pdf
IOS_CFG	Configuration Profile Key Reference Author(s) Apple Inc. Version 2017 Date 2017 Location https://developer.apple.com/library/content/featuredarticles/iPhoneConfigurationProfileRef/Introduction/Introduction.html

IOS_LOGS	Profiles and Logs Date received 2017-03-20 Location https://developer.apple.com/bug-reporting/profiles-and-logs/?platforms=ios
IOS_MDM	Mobile Device Management Protocol Reference Date 2016-01-20 Location https://developer.apple.com/go/?id=mobile-device-management-protocol-reference File name agd/MDM_Protocol_Reference.pdf
iOSDeployRef	iOS Deployment Reference Version EN019-00364/2017-03 Date received 2017 Location https://itunes.apple.com/us/book/ios-deployment-reference/id917468024?mt=11 File name agd/iOS_DeploymentReference.pdf
iPad_UG	iPad User Guide Version iOS 11.0 Date received 2017-11-15 File name agd/iPad_User_Guide.pdf
iPhone_UG	iPhone User Guide Version iOS 11.0 Date received 2017-11-15 File name agd/iPhone_User_Guide.pdf
IPLKEYREF	Information Property List Key Reference Date 2017-03-21 Location https://developer.apple.com/library/ios/documentation/General/Reference/InfoPlistKeyReference/Introduction/Introduction.html
KEYCHAINPG	Keychain Services Programming Guide Date 2016-09-13 Location https://developer.apple.com/library/ios/documentation/Security/Conceptual/keychainServConcepts/01introduction/introduction.html File name agd/keychainServConcepts.pdf
LC	Apple iOS 11.0 Life Cycle Version 1.1 Date 2018-03-30 File name alc/Apples_iOS_11_LifeCycle.pdf
OTA_CFG	Over-the-Air Profile Delivery and Configuration Date 2014-04-17 Location https://developer.apple.com/library/ios/documentation/NetworkingInternet/Conceptual/iPhoneOTAConfiguration/Introduction/Introduction.html File name agd/iPhoneOTAConfiguration.pdf

PP_MD_V3.1 Protection Profile for Mobile Device Fundamentals, Version 3.1

Version 3.1
Date received 2017-06-16
Location https://www.niap-ccevs.org/pp/pp_md_v3.1.pdf

PP_WLAN_CLI_EP_V1.0 Extended Package for Wireless LAN Client Version 1.0

Version 1.0
Date February 11, 2016
Location https://www.niap-ccevs.org/pp/pp_wlan_cli_ep_v1.0.pdf
File name [ase/pp_wlan_cli_ep_v1.0.pdf](#)

SecCodeGuide Secure Coding Guide

Date 2016-09-13
Location <https://developer.apple.com/library/content/documentation/Security/Conceptual/SecureCodingGuide/Introduction.html>

SECFWREF Security Framework Reference

Date received 2017
Location <https://developer.apple.com/reference/security>

ST 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

Version 1.01
Date 2018-03-30
File name [ase/st-vid10851_st-PUBLIC.pdf](#)

A.2 Glossary

Augmentation

The addition of one or more requirement(s) to a package.

Authentication data

Information used to verify the claimed identity of a user.

Authorised user

A user who may, in accordance with the SFRs, perform an operation.

Class

A grouping of CC families that share a common focus.

Component

The smallest selectable set of elements on which requirements may be based.

Connectivity

The property of the TOE which allows interaction with IT entities external to the TOE. This includes exchange of data by wire or by wireless means, over any distance in any environment or configuration.

Dependency

A relationship between components such that if a requirement based on the depending component is included in a PP, ST or package, a requirement based on the component that is depended upon must normally also be included in the PP, ST or package.

Deterministic RNG (DRNG)

An RNG that produces random numbers by applying a deterministic algorithm to a randomly selected seed and, possibly, on additional external inputs.

Element

An indivisible statement of security need.

Entropy

The entropy of a random variable X is a mathematical measure of the amount of information gained by an observation of X.

Evaluation

Assessment of a PP, an ST or a TOE, against defined criteria.

Evaluation Assurance Level (EAL)

An assurance package, consisting of assurance requirements drawn from CC Part 3, representing a point on the CC predefined assurance scale.

Evaluation authority

A body that implements the CC for a specific community by means of an evaluation scheme and thereby sets the standards and monitors the quality of evaluations conducted by bodies within that community.

Evaluation scheme

The administrative and regulatory framework under which the CC is applied by an evaluation authority within a specific community.

Exact conformance

a subset of Strict Conformance as defined by the CC, is defined as the ST containing all of the requirements in the Security Requirements section of the PP, and potentially requirements from Appendices of the PP. While iteration is allowed, no additional requirements (from the CC parts 2 or 3) are allowed to be included in the ST. Further, no requirements in the Security Requirements section of the PP are allowed to be omitted.

Extension

The addition to an ST or PP of functional requirements not contained in Part 2 and/or assurance requirements not contained in Part 3 of the CC.

External entity

Any entity (human or IT) outside the TOE that interacts (or may interact) with the TOE.

Family

A grouping of components that share a similar goal but may differ in emphasis or rigour.

Formal

Expressed in a restricted syntax language with defined semantics based on well-established mathematical concepts.

Guidance documentation

Documentation that describes the delivery, preparation, operation, management and/or use of the TOE.

Identity

A representation (e.g. a string) uniquely identifying an authorised user, which can either be the full or abbreviated name of that user or a pseudonym.

Informal

Expressed in natural language.

Object

A passive entity in the TOE, that contains or receives information, and upon which subjects perform operations.

Operation (on a component of the CC)

Modifying or repeating that component. Allowed operations on components are assignment, iteration, refinement and selection.

Operation (on an object)

A specific type of action performed by a subject on an object.

Operational environment

The environment in which the TOE is operated.

Organisational Security Policy (OSP)

A set of security rules, procedures, or guidelines imposed (or presumed to be imposed) now and/or in the future by an actual or hypothetical organisation in the operational environment.

Package

A named set of either functional or assurance requirements (e.g. EAL 3).

PP evaluation

Assessment of a PP against defined criteria.

Protection Profile (PP)

An implementation-independent statement of security needs for a TOE type.

Random number generator (RNG)

A group of components or an algorithm that outputs sequences of discrete values (usually represented as bit strings).

Refinement

The addition of details to a component.

Role

A predefined set of rules establishing the allowed interactions between a user and the TOE.

Secret

Information that must be known only to authorised users and/or the TSF in order to enforce a specific SFP.

Secure state

A state in which the TSF data are consistent and the TSF continues correct enforcement of the SFRs.

Security attribute

A property of subjects, users (including external IT products), objects, information, sessions and/or resources that is used in defining the SFRs and whose values are used in enforcing the SFRs.

Security Function Policy (SFP)

A set of rules describing specific security behaviour enforced by the TSF and expressible as a set of SFRs.

Security objective

A statement of intent to counter identified threats and/or satisfy identified organisation security policies and/or assumptions.

Security Target (ST)

An implementation-dependent statement of security needs for a specific identified TOE.

Seed

Value used to initialize the internal state of an RNG.

Selection

The specification of one or more items from a list in a component.

Semiformal

Expressed in a restricted syntax language with defined semantics.

ST evaluation

Assessment of an ST against defined criteria.

Subject

An active entity in the TOE that performs operations on objects.

Target of Evaluation (TOE)

A set of software, firmware and/or hardware possibly accompanied by guidance.

TOE evaluation

Assessment of a TOE against defined criteria.

TOE resource

Anything useable or consumable in the TOE.

TOE Security Functionality (TSF)

A set consisting of all hardware, software, and firmware of the TOE that must be relied upon for the correct enforcement of the SFRs.

Transfers outside of the TOE

TSF mediated communication of data to entities not under control of the TSF.

True RNG (TRNG)

A device or mechanism for which the output values depend on some unpredictable source (noise source, entropy source) that produces entropy.

Trusted channel

A means by which a TSF and a remote trusted IT product can communicate with necessary confidence.

Trusted path

A means by which a user and a TSF can communicate with necessary confidence.

TSF data

Data created by and for the TOE, that might affect the operation of the TOE.

TSF Interface (TSFI)

A means by which external entities (or subjects in the TOE but outside of the TSF) supply data to the TSF, receive data from the TSF and invoke services from the TSF.

User

See external entity

User data

Data created by and for the user, that does not affect the operation of the TSF.