macOS Catalina 10.15 Security Target

Acumen Security, LLC.

Document Version: 2.0
Date: September 18, 2020

Table of Contents

Revision History

Version	Date	Description
0.1	10/9/2019	Initial Draft
0.2	10/16/2019	Updating TSS sections
0.3	10/18/2019	Completion of TSS.
0.4	11/06/2019	Updates based vendor feedback
0.5	11/13/2019	Updates based on further reviews of ST
0.6	12/2/2019	Updates made based on testing
0.7	1/06/2020	Updates made based on vendor feedback
0.8	2/3/2020	Updates made on processor specifications and CAVPs.
0.9	4/21/2020	Addressing validator comments
1.0	5/11/2020	Updates made to SFRs.
1.1	6/2/2020	Updates made to FMT_SMF, TSS Audit Logs description and Processor specifications.
1.2	7/27/2020	Removed macOS Built in Store from FPT_TUD.
1.3	7/28/2020	Updating TDs and TSS sections.
1.4	8/4/2020	Removed macOS Built in store from FPT_TUD for Application update.
1.5	8/11/2020	Minor updates.
1.6	8/21/2020	Updated CAVP Table.
1.7	8/26/2020	Addressing vendor comments
1.8	9/16/2020	Addressing validator comments
1.9	9/17/2020	Updated AGD version
2.0	9/18/2020	Updated final AGD version

1 Security Target Introduction

1.1 Security Target and TOE Reference

This section provides information needed to identify and control this ST and its TOE.

Category	Identifier
ST Title	macOS Catalina 10.15 Security Target
ST Version	2.0
ST Date	September 18, 2020
ST Author	Acumen Security, LLC.
TOE Identifier	macOS Catalina
TOE Software Version	10.15.6
TOE Developer	Apple Inc.
Key Words	Operating System, GPOS

Table 1 TOE/ST Identification

1.2 TOE Overview

The TOE is a general-purpose operating system (GPOS) which runs on Mac mini, MacBook Air, MacBook Pro and Mac Pro iPad which include the T2 chip. The macOS Catalina is a Unix-based graphical operating system. macOS core is a POSIX compliant operating system built on top of the XNU kernel with standard Unix facilities available from the command line interface.

1.2.1 TOE Product Type

The TOE type is a general-purpose operating system. It satisfies all the criterion to meet the Protection Profile for General Purpose Operating Systems Version 4.2.1 [OS PP v4.2.1].

1.3 TOE Description

1.3.1 Evaluated Configuration

The TOE includes the operating system macOS Catalina 10.15.6 (Build 19G73) and the security processor (T2) (SEPOS build 17P5300).

The Apple T2 Security Chip is custom silicon for the Mac. It contains the Secure Enclave coprocessor which provides security related functionality that secures Touch ID data and provides the foundation for new encrypted storage and secure boot capabilities. Each of the TOE platforms includes both the Apple T2 Security Chip (T2) and an Intel CPU where the TOE runs.

NOTE: The TOE boundary would include the T2 chip and the Intel CPU.

The TOE will comply with [Use Case 1] End User Devices as outlined in Section 1.4 of the GPOS PP.

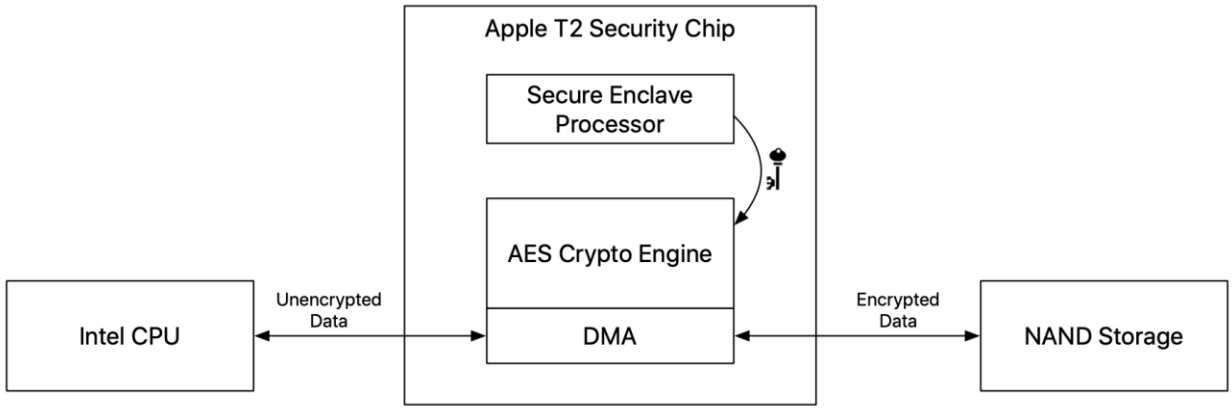

Figure 1: Apple T2 Security Chip and SEP

1.3.2 Devices covered by this evaluation

Micro-architecture	Processor - Intel Core	Device Family	Hardware Reference	Model	Marketing Release Name
Amber Lake	Intel i5-8210Y	MacBook Air	MacBookAir 8,2	A1932	2019
Amber Lake	Intel i5-8210Y	MacBook Air	MacBookAir 8,1	A1932	Late 2018
Coffee Lake	Intel i5-8257U	MacBook Pro	MacBook Pro16,3	A2289	2020, 13-inch
Coffee Lake	Intel i5-8257U	MacBook Pro	MacBookPro15,4	A2159	2019 13-inch (Touch Bar, 2TB 3)
Coffee Lake	Intel i5-8259U	MacBook Pro	MacBookPro15,2	A1989	Mid 2018, 13-inch (Touch Bar)
Coffee Lake	Intel i5-8279U	MacBook Pro	MacBookPro15,2	A1989	2019, 13-inch (Touch Bar)
Coffee Lake	Intel i5-8279U	MacBook Pro	MacBookPro15,2	A1989	Mid 2018, 13-inch (Touch Bar)
Coffee Lake	Intel i5-8500B	Mac mini	Macmini8,1	A1993	2018

Coffee Lake	Intel i7-8557U	MacBook Pro	MacBook Pro16,3	A2289	2020, 13-inch
Coffee Lake	Intel i7-8557U	MacBook Pro	MacBookPro15,4	A2159	2019 13-inch (Touch Bar, 2TB 3)
Coffee Lake	Intel i7-8559U	MacBook Pro	MacBookPro15,2	A1989	Mid 2018, 13-inch (Touch Bar)
Coffee Lake	Intel i7-8569U	MacBook Pro	MacBookPro15,2	A1989	2019, 13-inch (Touch Bar)
Coffee Lake	Intel i7-8700B	Mac mini	Macmini8,1	A1993	2018
Coffee Lake	Intel i7-8750H	MacBook Pro	MacBookPro15,1	A1990	Mid 2018, 15-inch (Touch Bar)
Coffee Lake	Intel i7-8850H	MacBook Pro	MacBookPro15,3	A1990	Mid 2018, 15-inch (Touch Bar)
Coffee Lake	Intel i7-9750H	MacBook Pro	MacBookPro15,1	A1990	2019, 15-inch (Touch Bar)
Coffee Lake	Intel i7-9750H	MacBook Pro	MacBookPro16,1	A2141	2019, 16-inch
Coffee Lake	Intel i9-8950HK	MacBook Pro	MacBookPro15,1	A1990	Mid 2018, 15-inch (Touch Bar)
Coffee Lake	Intel i9-8950HK	MacBook Pro	MacBookPro15,3	A1990	Mid 2018, 15-inch (Touch Bar)

Coffee Lake	Intel i9-9880H	MacBook Pro	MacBookPro15,1	A1990	2019, 15-inch (Touch Bar)
Coffee Lake	Intel i9-9880H	MacBook Pro	MacBookPro15,3	A1990	2019, 15-inch (Touch Bar)
Coffee Lake	Intel i9-9880H	MacBook Pro	MacBookPro16,1	A2141	2019, 16-inch
Coffee Lake	Intel i9-9980HK	MacBook Pro	MacBookPro15,1	A1990	2019, 15-inch (Touch Bar)
Coffee Lake	Intel i9-9980HK	MacBook Pro	MacBookPro15,3	A1990	2019, 15-inch (Touch Bar)
Coffee Lake	Intel i9-9980HK	MacBook Pro	MacBookPro16,2	A2141	2019, 16-inch
Ice lake	Intel i5-1030NG7	MacBook Air	MacBookAir9,1	A2179	2020
Ice Lake	Intel i5-1038NG7	MacBook Pro	MacBook Pro16,2	A2251	2020, 13-inch
Ice Lake	Intel i7-1060NG7	MacBook Air	MacBookAir9,1	A2179	2020
Ice Lake	Intel i7-1068NG7	MacBook Pro	MacBook Pro16,2	A2251	2020, 13-inch
Skylake	Intel Xeon W-2140B	iMac Pro	iMacPro1,1	A1862	iMac Pro, Late 2017
Skylake	Intel Xeon W-2150B	iMac Pro	iMacPro1,1	A1862	iMac Pro, Late 2017
Skylake	Intel Xeon W-2170B	iMac Pro	iMacPro1,1	A1862	iMac Pro, Late 2017
Skylake	Intel Xeon W-2191B	iMac Pro	iMacPro1,1	A1862	iMac Pro, Late 2017
Cascade Lake	Intel Xeon W-3223	Mac Pro	MacPro7,1	A1991	2019
Cascade Lake	Intel Xeon W-3223	Mac Pro(rack)	MacPro7,1	A2304	2019

Cascade Lake	Intel Xeon W-3235	Mac Pro	MacPro7,1	A1991	2019
Cascade Lake	Intel Xeon W-3235	Mac Pro(rack)	MacPro7,1	A2304	2019
Cascade Lake	Intel Xeon W-3245	Mac Pro	MacPro7,1	A1991	2019
Cascade Lake	Intel Xeon W-3245	Mac Pro(rack)	MacPro7,1	A2304	2019
Cascade Lake	Intel Xeon W-3265M	Mac Pro	MacPro7,1	A1991	2019
Cascade Lake	Intel Xeon W-3265M	Mac Pro(rack)	MacPro7,1	A2304	2019
Cascade Lake	Intel Xeon W-3275M	Mac Pro	MacPro7,1	A1991	2019
Cascade Lake	Intel Xeon W-3275M	Mac Pro(rack)	MacPro7,1	A2304	2019

Table 2 Platform specifications

1.3.3 Physical Boundaries

The TOE is a general-purpose operating system running on the listed platforms as indicated in Table 2.

1.3.4 Logical Scope of the TOE

The TOE implements the following security functional requirements from [GPOSPP] as listed below:

1.3.4.1 Audit Data Generation (FAU)

The TOE generates audit events for all start-up and shut-down functions, and all auditable events as specified in GPOS PP. Audit events are generated for the following audit functions:

- Start-up and shut-down of the audit functions
- Authentication events (Success/Failure)
- Use of privileged/special rights events (Successful and unsuccessful security, audit, and configuration changes)
- Privilege or role escalation events (Success/Failure)

Each audit record contains the date and time of the event, type of event, subject identity (if applicable), and outcome (success or failure) of the event.

1.3.4.2 Cryptographic Support (FCS)

Each of these cryptographic algorithms have been validated for conformance to the requirements specified in their respective standards, as identified below:

Algorithm	Standard	CAVP Certificates
AES	• AES-CBC (as defined in NIST SP 800-38A)	**CoreCrypto User Certs:** A7 (c_asm), A8 (c_ltc), A11 (c_glad), A19 (asm_aesni), A21 (c_aesni),

Algorithm	Standard	CAVP Certificates
		A25 (asm_x86) **CoreCrypto Kernel Certs:** A15 (c_asm), A20 (asm_aesni), A23 (c_aesni), A24 (asm_x86), A25 (asm_x86)
	• AES-GCM (as defined in NIST SP 800-38D)	**CoreCrypto User Certs:** A7 (c_asm), A8 (c_ltc), A10 (vng_asm), A21 (c_aesni), A31 (vng_aesni) **CoreCrypto Kernel Certs:** A13 (vng_asm), A28 (vng_aesni)
RSA	• FIPS PUB 186-4 Digital Signature Standard (DSS), Appendix B.3.	**CoreCrypto User Certs:** A8 (c_ltc), A22 (c_avx), A27 (c_sse3), A33 (c_avx2) **CoreCrypto Kernel Certs:** A26 (c_avx2), A30 (c_avx), A34 (c_sse3)
ECDSA	FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4	**CoreCrypto User Certs:** A8 (c_ltc), A22 (c_avx), A27 (c_sse3), A33 (c_avx2) **CoreCrypto Kernel Certs:** A26 (c_avx2), A30 (c_avx), A34 (c_sse3)
KAS/CVL ECC	• NIST Special Publication 800-56A	**CoreCrypto User Certs:** A8 (c_ltc)
HMAC	• Keyed-hash message authentication services in conforming to FIPS Pub 198-1 The Keyed-Hash Message Authentication Code and FIPS Pub 180-4 Secure Hash Standard	**CoreCrypto User Certs:** A8 (c_ltc), A22 (c_avx), A27 (c_sse3), A29 (vng_intel), A33 (c_avx2)

Algorithm	Standard	CAVP Certificates
		CoreCrypto Kernel Certs: A26 (c_avx2), A30 (c_avx), A32 (vng_intel), A34 (c_sse3)
SHS	• NIST FIPS Pub 180-4.	**CoreCrypto User Certs:** A8 (c_ltc), A22 (c_avx) , A27 (c_sse3), A29 (vng_intel), A33 (c_avx2) **CoreCrypto Kernel Certs**: A26 (c_avx2), A30 (c_avx), A32 (vng_intel), A34 (c_sse3)
DRBG	• CTR_DRBG (AES)	**CoreCrypto User Certs:** A7 (c_asm) A8 (c_ltc), A10 (vng_asm), A21 (c_aesni), A31 (vng_aesni)**CoreCrypto Kernel Certs:** A15(c_asm), A23 (c_aesni), A13 (vng_asm), A28 (vng_aesni)
CVL TLS v1.2	• KDF 800-108	**CoreCrypto User Certs:** A8 (c_ltc), A22 (c_avx), A27 (c_sse3) **CoreCrypto Kernel Certs:** A34 (c_sse3)

Table 3 CAVP Algorithm Testing References

1.3.4.3 User Data Protection (FDP)

The TOE implements access controls which prevents unprivileged users from accessing files and directories owned by other users. The TOE provides an interface which allows VPN client to protect all IP traffic.

1.3.4.4 Identification and Authentication (FIA)

All users must be authenticated to the TOE prior to carrying out any management actions. The TOE supports password-based authentication, authentication based on username and a PIN that releases asymmetric key stored in OE-protected storage and X509 certificate-based authentication. The TOE will lock out user accounts after a defined number of unsuccessful authentication attempts have been met.

1.3.4.5 Security Management (FMT)

The TOE can perform management functions. The administrator has full access to carry-out all management functions and the user have limited privilege.

1.3.4.6 Protection of the TSF (FPT)

The TOE implements the following protection of TSF data:

- Access Controls
- Randomize process address space memory locations with 16 bit of entropy.
- Stack buffer overflow protection is used
- Verification of integrity of the boot-chain and operating system executable code and application executable code.
- Trusted software updates using digital signatures.

1.3.4.7 Trusted Path/Channels (FTP)

The TOE supports TLS v1.2 for trusted channel and trusted path communications.

1.3.4.8 TOE Access (FTA)

Before establishing a user session, the TOE will display an advisory warning message regarding unauthorized use of the OS.

1.4 Excluded Functionality

The following interfaces are not included as part of the evaluated configuration:

Functions	Exclusion discussion
Two-Factor Authentication	Two-factor authentication is an extra layer of security for an Apple ID used in the Apple store, iCloud and other Apple services. It is designed to enhance the security on these on-line Apple accounts. This feature is outside the scope of the evaluation.
Bonjour	Bonjour is Apple's standards-based, zero configuration network protocol that lets devices find services on a network. This feature is outside the scope of the evaluation.
VPN Split Tunnel	VPN split tunnel is not included in the evaluation, and must be disabled in the mobile device configurations meeting the requirements of this CC evaluation.
Siri Interface	The Siri interface supports some commands related to configuration settings. This feature is not included in the evaluation and must be disabled in the mobile device configurations that meet the requirements of this CC evaluation.

Table 4 Excluded Functionality

1.5 TOE Documentation

The following documents are available in PDF formats.

Documentation	File Format	Date
macOS Catalina Security Target v2.0	PDF	9 18 2020
Apple macOS Catalina Common Criteria Guidance Document v1.7	PDF	9 18 2020

Table 5 TOE Documentation

1.6 Other References

- Protection Profile for General Purpose Operating Systems, Version 4.2.1. [GPOSPP]

2 Conformance Claims

2.1 CC Conformance

This TOE is conformant to:

- Common Criteria for Information Technology Security Evaluations Part 1, Version 3.1, Revision 5, April 2017
- Common Criteria for Information Technology Security Evaluations Part 2, Version 3.1, Revision 5, April 2017: Part 2 extended
- Common Criteria for Information Technology Security Evaluations Part 2, Version 3.1, Revision 5, April 2017: Part 3 extended

2.2 Protection Profile Conformance

This TOE is conformant to:

- Protection Profile for General Purpose Operating Systems, Version 4.2.1 [GPOSPP]

2.3 Conformance Rationale

This Security Target provides exact conformance to "[GPOSPP]". The security problem definition, security objectives and security requirements in this Security Target are all taken from the Protection Profile performing only operations defined there.

2.3.1 Technical Decisions

All NIAP Technical Decisions (TDs) issued to date that are applicable to [GPOSPP] have been addressed. The following table identifies all applicable TDs:

Identifier	Applicable	Exclusion Rationale (if applicable)
TD0525: Updates to Certificate Revocation (FIA_X509_EXT.1)	Yes	
TD0496: GPOS PP adds allow-with statement for VPN Client V2.1	Yes	
TD0493: X.509v3 certificates when using digital signatures for Boot Integrity	Yes	
TD0463 - Clarification for FPT_TUD_EXT	Yes	
TD0441 - Updated TLS Ciphersuites for OS PP	No	The following ciphersuites are not being claimed: FCS_TLSC_EXT.1.1 in the OS PP omits the TLS_RSA_WITH_AES_256_CBC_SHA, TLS_RSA_WITH_AES_128_GCM_SHA256, and TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 ciphersuites.
TD0386 – Platform-Provided Verification of Update	Yes	
TD0365 – FCS_CKM_EXT.4 selections	Yes	

Table 6 GPOS Technical Decisions

3 Security Problem Definition

The security problem definition has been taken from [GPOSPP] and is reproduced here for the convenience of the reader. The security problem is described in terms of the threats that the TOE is expected to address, assumptions about the operational environment, and any organizational security policies that the TOE is expected to enforce.

3.1 Threats

The following threats are drawn directly from the [GPOSPP].

ID	Threat
T.NETWORK_ATTACK	An attacker is positioned on a communications channel or elsewhere on the network infrastructure. Attackers may engage in communications with applications and services running on or part of the OS with the intent of compromise. Engagement may consist of altering existing legitimate communications.
T.NETWORK_EAVESDROP	An attacker is positioned on a communications channel or elsewhere on the network infrastructure. Attackers may monitor and gain access to data exchanged between applications and services that are running on or part of the OS.
T.LOCAL_ATTACK	An attacker may compromise applications running on the OS. The compromised application may provide maliciously formatted input to the OS through a variety of channels including unprivileged system calls and messaging via the file system.
T.LIMITED_PHYSICAL_ACCESS	An attacker may attempt to access data on the OS while having a limited amount of time with the physical device.

Table 7 Threats

3.2 Assumptions

The following assumptions are drawn directly from the [GPOSPP].

ID	Assumption
A.PLATFORM	The OS relies upon a trustworthy computing platform for its execution. This underlying platform is out of scope of this PP.
A.PROPER_USER	The user of the OS is not willfully negligent or hostile, and uses the software in compliance with the applied enterprise security policy. At the same time, malicious software could act as the user, so requirements which confine malicious subjects are still in scope.
A.PROPER_ADMIN	The administrator of the OS is not careless, willfully negligent or hostile, and administers the OS within compliance of the applied enterprise security policy.

Table 8 Assumptions

3.3 Organizational Security Policies

The [GPOSPP]do not define any OSPs.

4 Security Objectives

The security objectives for the TOE have been taken from [GPOSPP] and are reproduced here for the convenience of the reader.

4.1 Security Objectives for the TOE

The following subsections describe objectives for the TOE.

ID	Objective for the Operation Environment
O.ACCOUNTABILITY	Conformant OSes ensure that information exists that allows administrators to discover unintentional issues with the configuration and operation of the operating system and discover its cause. Gathering event information and immediately transmitting it to another system can also enable incident response in the event of system compromise.
O.INTEGRITY	Conformant OSes ensure the integrity of their update packages. OSes are seldom if ever shipped without errors, and the ability to deploy patches and updates with integrity is critical to enterprise network security. Conformant OSes provide execution environment-based mitigations that increase the cost to attackers by adding complexity to the task of compromising systems.
O.MANAGEMENT	To facilitate management by users and the enterprise, conformant OSes provide consistent and supported interfaces for their security-relevant configuration and maintenance. This includes the deployment of applications and application updates through the use of platform-supported deployment mechanisms and formats, as well as providing mechanisms for configuration and application execution control.
O.PROTECTED_STORAGE	To address the issue of loss of confidentiality of credentials in the event of loss of physical control of the storage medium, conformant OSes provide data-at-rest protection for credentials. Conformant OSes also provide access controls which allow users to keep their files private from other users of the same system.
O.PROTECTED_COMMS	To address both passive (eavesdropping) and active (packet modification) network attack threats, conformant OSes provide mechanisms to create trusted channels for CSP and sensitive data. Both CSP and sensitive data should not be exposed outside of the platform.

Table 9 Security Objectives for the TOE

4.2 Security Objectives for the Operational Environment

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

ID	Objective for the Operation Environment
OE.PLATFORM	The OS relies on being installed on trusted hardware.

OE.PROPER_USER	The user of the OS is not willfully negligent or hostile, and uses the software within compliance of the applied enterprise security policy. Standard user accounts are provisioned in accordance with the least privilege model. Users requiring higher levels of access should have a separate account dedicated for that use.
OE.PROPER_ADMIN	The administrator of the OS is not careless, willfully negligent or hostile, and administers the OS within compliance of the applied enterprise security policy.

Table 10 Objectives for the Operational Environment

4.3 Rationale for Security Objectives

The following section describes how the assumptions, threats, and organizational security policies map to the security objectives.

Threat, Assumption, or OSP	Security Objectives	Rationale
T.NETWORK_ATTACK	O.PROTECTED_COMMS, O.INTEGRITY, O.MANAGEMENT O.ACCOUNTABILITY	The threat T.NETWORK_ATTACK is countered by O.PROTECTED_COMMS as this provides for integrity of transmitted data. The threat T.NETWORK_ATTACK is countered by O.INTEGRITY as this provides for integrity of software that is installed onto the system from the network. The threat T.NETWORK_ATTACK is countered by O.MANAGEMENT as this provides for the ability to configure the OS to defend against network attack. The threat T.NETWORK_ATTACK is countered by O.ACCOUNTABILITY as this provides a mechanism for the OS to report behavior that may indicate a network attack has occurred.
T.NETWORK_EAVESDROP	O.PROTECTED_COMMS, O.MANAGEMENT	The threat T.NETWORK_EAVESDROP is countered by O.PROTECTED_COMMS as this provides for confidentiality of transmitted data. The threat T.NETWORK_EAVESDROP is countered by O.MANAGEMENT as this provides for the ability to configure the OS to protect the

		confidentiality of its transmitted data.
T.LOCAL_ATTACK	O.INTEGRITY O.ACCOUNTABILITY	The objective O.INTEGRITY protects against the use of mechanisms that weaken the TOE with regard to attack by other software on the platform. The objective O.ACCOUNTABILITY protects against local attacks by providing a mechanism to report behavior that may indicate a local attack is occurring or has occurred.
T.LIMITED_PHYSICAL_ACCESS	O.PROTECTED_STORAGE	The objective O.PROTECTED_STORAGE protects against unauthorized attempts to access physical storage used by the TOE.
A.PLATFORM	OE.PLATFORM	The operational environment objective OE.PLATFORM is realized through A.PLATFORM.
A.PROPER_USER	OE.PROPER_USER	The operational environment objective OE.PROPER_USER is realized through A.PROPER_USER.
A.PROPER_ADMIN	OE.PROPER_ADMIN	The operational environment objective OE.PROPER_ADMIN is realized through A.PROPER_ADMIN.

Table 11 Rationale for Security Objectives

5 Extended Security Functional Components

Requirements	Descriptions
FCS_RBG_EXT.1	Random Bit Generation
FCS_STO_EXT.1	Storage of Sensitive Data
FCS_TLSC_EXT.1	TLS Client Protocol
FCS_TLSC_EXT.2	TLS Client Curves Allowed
FDP_ACF_EXT.1	Access Controls for Protecting User Data
FIA_X509_EXT.1	X.509 Certificate Validation
FIA_X509_EXT.2	X.509 Certificate Authentication
FMT_MOF_EXT.1	Management of security functions behavior
FMT_SMF_EXT.1	Specification of Management Functions
FPT_ACF_EXT.1	Access controls
FPT_ASLR_EXT.1	Address Space Layout Randomization
FPT_SBOP_EXT.1	Stack Buffer Overflow Protection
FPT_TST_EXT.1	Boot Integrity
FPT_TUD_EXT.1	Trusted Update
FPT_TUD_EXT.2	Trusted Update for Application Software
FTP_ITC_EXT.1	Trusted channel communication

Table 12 Extended Security Functional Components

5.1 Extended Security Functional Components Rationale

The definition of all SFRs with the appendix of "_EXT" is supplied by the protection profile. All extended security functional components are derived directly from the [OS PP v4.2.1] and applied verbatim.

6 Security Requirements

This section identifies the Security Functional Requirements for the TOE. The Security Functional Requirements included in this section are derived from Part 2 of the Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 5, dated: April 2017 and all international interpretations.

Requirements	Descriptions
FAU_GEN.1	Audit Data Generation (Refined)
FCS_CKM.1	Cryptographic Key Generation (Refined)
FCS_CKM.2	Cryptographic Key Establishment (Refined)
FCS_CKM_EXT.4	Cryptographic Key Destruction
FCS_COP.1(1)	Cryptographic Operation - Encryption/Decryption (Refined)
FCS_COP.1(2)	Cryptographic Operation - Hashing (Refined)
FCS_COP.1(3)	Cryptographic Operation - Signing (Refined)
FCS_COP.1(4)	Cryptographic Operation - Keyed-Hash Message Authentication (Refined)
FCS_RBG_EXT.1	Random Bit Generation
FCS_STO_EXT.1	Storage of Sensitive Data
FCS_TLSC_EXT.1	TLS Client Protocol
FCS_TLSC_EXT.2	TLS Client Protocol
FCS_TLSC_EXT.4	TLS Client Protocol
FDP_ACF_EXT.1	Access Controls for Protecting User Data
FIA_AFL.1	Authentication Failure Management (Refined
FIA_UAU.5	Multiple Authentication Mechanisms (Refined)
FIA_X509_EXT.1	X.509 Certificate Validation
FIA_X509_EXT.2	X.509 Certificate Authentication
FMT_MOF_EXT.1	Management of security functions behavior
FMT_SMF_EXT.1	Specification of Management Functions
FPT_ACF_EXT.1	Access controls
FPT_ASLR_EXT.1	Address Space Layout Randomization
FPT_SBOP_EXT.1	Stack Buffer Overflow Protection
FPT_TST_EXT.1	Boot Integrity
FPT_TUD_EXT.1	Trusted Update
FPT_TUD_EXT.2	Trusted Update for Application Software
FTP_ITC_EXT.1	Trusted channel communication
FTP_TRP.1	Trusted Path
FTA_TAB.1	Default TOE access banners

Table 13 SFRs

6.1 Conventions

The CC defines operations on Security Functional Requirements: assignments, selections, assignments within selections and refinements. This document uses the following font conventions to identify the operations defined by the CC:

- Assignment: Indicated with *italicized* text;
- Refinement: Indicated with **bold** text;
- Selection: Indicated with underlined text;
- Iteration: Indicated by appending the SFR name with a slash and unique identifier suggesting the purpose of the iteration, e.g. '/TLS' for an SFR relating to TLS functionality and/or a sequential number in parentheses, e.g. (1).
- Where operations were completed in the PP or EP itself, the formatting used in the PP or EP has been retained.

Extended SFRs are identified by having a label 'EXT' after the requirement name. Formatting conventions outside of operations matches the formatting specified within the PP or EP.

6.2 Security Functional requirements

6.2.1 Cryptographic Support (FCS)

6.2.1.1 FCS_CKM.1 Cryptographic Key Generation (Refined)

FCS_CKM.1.1 The **OS** shall generate **asymmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm: [

- ***RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3,***
- ***ECC schemes using "NIST curves" P-256, P-384 and [P-521] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4,***

] ~~and specified cryptographic key sizes [assignment: cryptographic key sizes] that meet the following: [assignment: list of standards]~~.

6.2.1.2 FCS_CKM.2 Cryptographic Key Establishment (Refined)

FCS_CKM.2.1 The OS shall **implement functionality to perform cryptographic key establishment** in accordance with a specified cryptographic key **establishment** method:

[*RSA-based key establishment schemes] that meets the following: [NIST Special Publication 800-56B, "Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography"]*

and [

- ***Elliptic curve-based key establishment schemes that meets the following: NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography",***

] ~~that meets the following: [assignment: list of standards]~~.

6.2.1.3 FCS_CKM_EXT.4 Cryptographic Key Destruction

FCS_CKM_EXT.4.1 The OS shall destroy cryptographic keys and key material in accordance with a specified cryptographic key destruction method [

- *For volatile memory, the destruction shall be executed by a [*
    - *single overwrite consisting of [zeros],*

]

- *For non-volatile memory that consists of*
    - *the invocation of an interface provided by the underlying platform that [*
    - *logically addresses the storage location of the key and performs a [single] overwrite consisting of [zeroes].]*

].

FCS_CKM_EXT.4.2 The OS shall destroy all keys and key material when no longer needed.

6.2.1.4 FCS_COP.1(1) Cryptographic Operation - Encryption/Decryption (Refined)

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

- ***AES-CBC (as defined in NIST SP 800-38A)***

] **and**

[

- ***AES-GCM (as defined in NIST SP 800-38D),***

] and cryptographic key sizes [*128-bit, 256-bit*] ~~that meet the following: [assignment: list of standards]~~.

6.2.1.5 FCS_COP.1(2) Cryptographic Operation - Hashing (Refined)

FCS_COP.1.1(2) The **OS** shall perform [*cryptographic hashing services*] in accordance with a specified cryptographic algorithm [*SHA-1 and* [

- *SHA-256,*
- *SHA-384,*
- *SHA-512,*

]] **and message digest sizes 160 bits and** [

- ***256 bits,***
- ***384 bits,***
- ***512 bits,***

] that meet the following: [*FIPS Pub 180-4*].

6.2.1.6 FCS_COP.1(3) Cryptographic Operation - Signing (Refined)

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

- ***RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 4,***
- ***ECDSA schemes using "NIST curves" P-256, P-384 and [P-521] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5***

] ~~and cryptographic key sizes [assignment: cryptographic algorithm] that meet the following: [assignment: list of standards]~~.

6.2.1.7 FCS_COP.1(4) Cryptographic Operation - Keyed-Hash Message Authentication (Refined)

FCS_COP.1.1(4) The **OS** shall perform [*keyed-hash message authentication services*] in accordance with a specified cryptographic algorithm [***SHA-1, SHA-256, SHA-384, SHA-512***] **with key sizes** [***128 and 256 bits used in HMAC]***] **and message digest** sizes **[*160 bits, 256 bits, 384 bits, 512 bits*]** that meet the following: [FIPS Pub 198-1 *The Keyed-Hash Message Authentication Code* and FIPS Pub 180-4 *Secure Hash Standard*].

6.2.1.8 FCS_RBG_EXT.1 Random Bit Generation

FCS_RBG_EXT.1.1 The OS shall perform all deterministic random bit generation (DRBG) services in accordance with NIST Special Publication 800-90A using [

- *CTR_DRBG (AES)*

].

FCS_RBG_EXT.1.2 The deterministic RBG used by the OS shall be seeded by an entropy source that accumulates entropy from a [

- *software-based noise source*

] with a minimum of [

- *256 bits*

] of entropy at least equal to the greatest security strength (according to NIST SP 800-57) of the keys and hashes that it will generate.

6.2.1.9 FCS_STO_EXT.1 Storage of Sensitive Data

FCS_STO_EXT.1.1 The OS shall implement functionality to encrypt sensitive data stored in non-volatile storage and provide interfaces to applications to invoke this functionality.

6.2.1.10 FCS_TLSC_EXT.1 TLS Client Protocol

FCS_TLSC_EXT.1.1 The OS shall implement TLS 1.2 (RFC 5246) supporting the following cipher suites: [

- *TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246*
- *TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246*
- *TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246*
- *TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289*
- *TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289*
- *TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*
- *TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289*

].

FCS_TLSC_EXT.1.2

The OS shall verify that the presented identifier matches the reference identifier according to RFC 6125.

FCS_TLSC_EXT.1.3

The OS shall only establish a trusted channel if the peer certificate is valid.

6.2.1.11 FCS_TLSC_EXT.2 TLS Client Protocol

FCS_TLSC_EXT.2.1 The **OS** shall present the Supported Groups Extension in the Client Hello with the following supported groups: [*secp256r1, secp384r1, secp521r1*].

6.2.1.12 FCS_TLSC_EXT.4 TLS Client Protocol

FCS_TLSC_EXT.4.1 The OS shall support mutual authentication using X.509v3 certificates.

6.2.2 User Data Protection (FDP)

6.2.2.1 FDP_ACF_EXT.1 Access Controls for Protecting User Data

FDP_ACF_EXT.1.1 The OS shall implement access controls which can prohibit unprivileged users from accessing files and directories owned by other users.

6.2.3 Identification and Authentication (FIA)

6.2.3.1 FIA_AFL.1 Authentication Failure Management (Refined)

FIA_AFL.1.1 The **OS** shall detect when [

- *an Administrator configurable positive integer within [1-50]*

] unsuccessful authentication attempts occur related to **events with [**

- ***authentication based on user name and password,***

].

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts for an account has been **met**, the **OS** shall: [***Account Lockout***].

6.2.3.2 FIA_UAU.5 Multiple Authentication Mechanisms (Refined)

FIA_UAU.5.1 The **OS** shall provide the following authentication mechanisms [

- ***authentication based on user name and password,***
- ***authentication based on user name and a PIN that releases an asymmetric key stored in OE-protected storage,***

] to support user authentication.

FIA_UAU.5.2 The **OS** shall authenticate any user's claimed identity according to the [

- *Authentication based on username and password: is performed for TOE-originated requests and with credentials stored by the OS.*

- *Authentication based on username and a PIN that releases an asymmetric key stored in OE-protected storage: local users can login to the TOE using a registered Smart Card. The asymmetric private key used for authentication is stored in the Smart Card.*
  ].

6.2.3.3 FIA_X509_EXT.1 X.509 Certificate Validation

FIA_X509_EXT.1.1

The OS shall implement functionality to validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation
- The certificate path must terminate with a trusted CA certificate
- The OS shall validate a certificate path by ensuring the presence of the basicConstraints extension, that the CA flag is set to TRUE for all CA certificates, and that any path constraints are met.
- The TSF shall validate that any CA certificate includes caSigning purpose in the key usage field
- The OS shall validate the revocation status of the certificate using [an OCSP TLS Status Request Extension (OCSP stapling) as specified in RFC 6066]
- The OS shall validate the extendedKeyUsage field according to the following rules:
  - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing Purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
  - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
  - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the EKU field.
  - S/MIME certificates presented for email encryption and signature shall have the Email Protection purpose (id-kp 4 with OID 1.3.6.1.5.5.7.3.4) in the EKU field.
  - OCSP certificates presented for OCSP responses shall have the OCSP Signing Purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the EKU field.
  - Server certificates presented for EST shall have the CMC Registration Authority (RA) purpose (id-kp-cmcRA with OID 1.3.6.1.5.5.7.3.28) in the EKU field. (conditional)

FIA_X509_EXT.1.2

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

6.2.3.4 FIA_X509_EXT.2 X.509 Certificate Authentication

FIA_X509_EXT.2.1

The OS shall use X.509v3 certificates as defined by RFC 5280 to support authentication for TLS and [*HTTPS*] connections.

6.2.4 Security Management (FMT)

6.2.4.1 FMT_MOF_EXT.1 Management of security functions behavior

FMT_MOF_EXT.1.1 The OS shall restrict the ability to perform the function indicated in the "Administrator" column in FMT_SMF_EXT.1.1 to the administrator.

6.2.4.2 FMT_SMF_EXT.1 Specification of Management Functions

FMT_SMF_EXT.1.1 The OS shall be capable of performing the following management functions:

Sr. No	Management Function	Administrator	User
1	Enable/disable [*screen lock*]	X	X
2	Configure [*screen lock*] inactivity timeout	X	X
3	Configure local audit storage capacity	X	-
4	Configure minimum password length	X	-
5	Configure minimum number of special characters in password	X	-
6	Configure minimum number of numeric characters in password	X	-
7	Configure minimum number of uppercase characters in password	X	-
8	Configure minimum number of lowercase characters in password	X	-
9	Configure lockout policy for unsuccessful authentication attempts through [*limiting number of attempts during a time period*]	X	-
10	Configure host-based firewall	X	-
11	Configure name/address of directory server with which to bind	-	-
12	Configure name/address of remote management server from which to receive management settings	-	-
13	Configure name/address of audit/logging server to which to send audit/logging records	X	-
14	Configure audit rules	X	-
15	Configure name/address of network time server	X	-
16	Enable/disable automatic software update	X	-
17	Configure WiFi interface	X	-
18	Enable/disable Bluetooth interface	X	X
19	Enable/disable [*no other external interfaces*]	-	-
20	[*no other management functions*]	-	-

Table 14 Specification of Management Functions

6.2.5 Protection of the TSF (FPT)

6.2.5.1 FPT_ACF_EXT.1 Access controls

FPT_ACF_EXT.1.1 The OS shall implement access controls which prohibit unprivileged users from modifying:

- Kernel and its drivers/modules
- Security audit logs
- Shared libraries
- System executables
- System configuration files
- [*none*]

FPT_ACF_EXT.1.2 The OS shall implement access controls which prohibit unprivileged users from reading:

- Security audit logs
- System-wide credential repositories
- [*none*]

6.2.5.2 FPT_ASLR_EXT.1 Address Space Layout Randomization

FPT_ASLR_EXT.1.1 The OS shall always randomize process address space memory locations with *[16 bits*] of entropy except for [*none].*

6.2.5.3 FPT_SBOP_EXT.1 Stack Buffer Overflow Protection

FPT_SBOP_EXT.1.1 The OS shall [*employ stack-based buffer overflow protections*].

6.2.5.4 FPT_TST_EXT.1 Boot Integrity

FPT_TST_EXT.1.1 The OS shall verify the integrity of the bootchain up through the OS kernel and [

- *no other executable code*

] prior to its execution through the use of [

- *a digital signature using a hardware-protected asymmetric key,*

].

6.2.5.5 FPT_TUD_EXT.1 Trusted Update

FPT_TUD_EXT.1.1 The OS shall provide the ability to check for updates to the OS software itself and shall use a digital signature scheme specified in FCS_COP.1(3) to validate the authenticity of the response.

FPT_TUD_EXT.1.2 The OS [shall cryptographically verify] updates to itself using a digital signature prior to installation using schemes specified in FCS_COP.1(3).

6.2.5.6 FPT_TUD_EXT.2 Trusted Update for Application Software

FPT_TUD_EXT.2.1 The OS shall provide the ability to check for updates to application software and shall use a digital signature scheme specified in FCS_COP.1(3) to validate the authenticity of the response.

FPT_TUD_EXT.2.2 The OS shall cryptographically verify the integrity of updates to applications using a digital signature specified by FCS_COP.1(3) prior to installation.

6.2.5.7 FAU_GEN.1 Audit Data Generation (Refined)

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

a. Start-up and shut-down of the audit functions;

b. All auditable events for the [not specified] level of audit; and [

c.

- Authentication events (Success/Failure);
- Use of privileged/special rights events (Successful and unsuccessful security, audit, and configuration changes);
- Privilege or role escalation events (Success/Failure);

]

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

a. Date and time of the event, type of event, subject identity (if applicable), and outcome (success or failure) of the event; and
b. For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [*none*].

6.2.6 Trusted path/channels (FTP)

6.2.6.1 FTP_ITC_EXT.1 Trusted channel communication

FTP_ITC_EXT.1.1 The OS shall use [

- *TLS as conforming to FCS_TLSC_EXT.1,*

] to provide a trusted communication channel between itself and authorized IT entities supporting the following capabilities: [*[application initiated TLS]*] that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from disclosure and detection of modification of the channel data.

6.2.6.2 FTP_TRP.1 Trusted Path

FTP_TRP.1.1 The **OS** shall provide a communication path between itself and [*local*] users that is logically distinct from other communication paths and provides assured identification of its endpoints and protection of the communicated data from modification and disclosure.

FTP_TRP.1.2 The **OS** shall permit [*local users*] to initiate communication via the trusted path.

FTP_TRP.1.3 The **OS** shall require use of the trusted path for [[*all remote administrative actions*]].

6.2.7 TOE Access (FTA)

6.2.7.1 FTA_TAB.1 Default TOE access banners

FTA_TAB.1 Before establishing a user session, the **OS** shall display an advisory warning message regarding unauthorized use of the OS.

6.3 TOE SFR Dependencies Rationale for SFRs

[GPOSPP] contain all the requirements claimed in this Security Target. As such, the dependencies are not applicable since the PP and EP have been approved.

6.4 Security Assurance Requirements

The TOE assurance requirements for this ST are taken directly from [GPOSPP] which are derived from Common Criteria Version 3.1, Revision 5. The assurance requirements are summarized in the table below.

Assurance Class	Components	Components Description
Development	ADV_FSP.1	Basic Functional Specification
Guidance Documentation	AGD_OPE.1	Operational User Guidance
	AGD_PRE.1	Preparative Procedures
Life-Cycle Support	ALC_CMC.1	Labeling of the TOE
	ALC_CMS.1	TOE CM Coverage
	ALC_TSU_EXT.1	Timely Security Updates
Tests	ATE_IND.1	Independent Testing – Conformance
Vulnerability Assessment	AVA_VAN.1	Vulnerability Survey

Table 15 Security Assurance Requirements

6.6 Rationale for Security Assurance Requirements

The functional specification describes the external interfaces of the TOE; such as the means for a user to invoke a service and the corresponding response of those services. The description includes the interface(s) that enforces a security functional requirement, the interface(s) that supports the enforcement of a security functional requirement, and the interface(s) that does not enforce any security functional requirements. The interfaces are described in terms of their purpose (general goal of the interface), method of use (how the interface is to be used), parameters (explicit inputs to and outputs from an interface that control the behavior of that interface), parameter descriptions (tells what the parameter is in some meaningful way), and error messages (identifies the condition that generated it, what the message is, and the meaning of any error codes). The development evidence also contains a tracing of the interfaces to the SFRs described in this ST.

6.7 Assurance Measures

The TOE satisfies the identified assurance requirements. This section identifies the Assurance Measures applied by Apple to satisfy the assurance requirements. The table below lists the details.

SAR Component	How the SAR will be met
ADV_FSP.1	The functional specification describes the external interfaces of the TOE; such as the means for a user to invoke a service and the corresponding response of those services. The description includes the interface(s) that enforces a security functional requirement, the interface(s) that supports the enforcement of a security functional requirement, and the interface(s) that does not enforce any security functional requirements. The interfaces are described in terms of their purpose (general goal of the interface), method of use (how the interface is to be used), parameters (explicit inputs to and outputs from an interface that control the behavior of that interface), parameter descriptions (tells what the parameter is in some meaningful way), and error messages (identifies the condition that generated it, what the message is, and the meaning of any error codes).
AGD_OPE.1	The Administrative Guide provides the descriptions of the processes and procedures of how the administrative users of the TOE can securely administer the TOE using the interfaces that provide the features and functions detailed in the guidance.
AGD_PRE.1	The Installation Guide describes the installation, generation, and startup procedures so that the users of the TOE can put the components of the TOE in the evaluated configuration.
ALC_CMC.1	The Configuration Management (CM) documents describe how the consumer identifies the evaluated TOE. The CM documents identify the configuration items, how those configuration items are uniquely identified, and the adequacy of the procedures that are used to control and track changes that are made to the TOE. This includes details on what changes are tracked and how potential changes are incorporated.
ALC_CMS.1	
ALC_TSU_EXT.1	The following page: https://support.apple.com/en-us/HT201222 contains all security updates for Apple's products; including the TOE components. This site will contain only available security updates.

	When a new update is released, macOS Catalina will automatically notify of the update through the notifications system on the host platform (i.e. the platforms display indicates that there is an update, allowing the user to click on the notification and initiate update process). This gives users direct access to install the update. Updates can also be manually updated from the following websites: • https://support.apple.com/en_US/downloads • https://support.apple.com/downloads To report security or privacy issues that affect Apple products or web servers, should contact product-security@apple.com. Submissions can use Apple's Product Security PGP key (https://support.apple.com/en-us/HT201214) to encrypt sensitive information that is sent by email. When the email is received, Apple will send an automatic email as acknowledgment. If this email is not received, please check the email address, and send again. For the protection of our customers, Apple generally does not disclose, discuss, or confirm security issues until a full investigation is complete and any necessary patches or releases are available. Apple distributes information about security issues in its products through security advisories. Users can also receive Apple security advisories through the security-announce mailing list.
ATE_IND.1	Apple will provide the TOE for testing.
AVA_VAN.1	Apple will provide the TOE for testing.

Table 16 TOE Security Assurance Measures

7 TOE Summary Specification

This chapter identifies and describes how the Security Functional Requirements identified above are met by the TOE.

TOE SFRs	Rationale
FAU_GEN.1 and FAU_GEN.2	Audit events are generated for the following audit functions: • Start-up and shut-down of the audit functions • Authentication events (Success/Failure) • Use of privileged/special rights events (Successful and unsuccessful security, audit, and configuration changes) • Privilege or role escalation events (Success/Failure) Each audit record contains the following information: Date and time of the event, type of event, subject identity (if applicable), and outcome (success or failure) of the event. The logging system captures messages across all levels of the system, and it stores the log data in memory and data store on disk. Audit events are only accessible by administrators. Audit events can be viewed via command-line . On POSIX systems, if the username is longer than 8 characters, the system uses the user ID to display. To access the audit logs, the TOE provides built-in utilities “audit”, “praudit”, and “auditreduce”. All audit records are BSM compliant, and any BSM Audit Tool could be used for viewing audit logs.
FCS_CKM.1	The TOE supports RSA key sizes of 2048 bits, and 3072 bits for key generation conforming to FIPS PUB 186-4 Digital Signature Standard (DSS), Appendix B.3. RSA keys are used for TLS sessions. The TOE acts as sender and receiver in the RSA key establishment scheme. The TOE supports Elliptical NIST curve sizes of P-256, P-384 and P-521 for key generation conforming to FIPS PUB 186-4 Digital Signature Standard (DSS)”, Appendix B.4. The Elliptic keys are used in support of ECDH key exchange. ECDH public and private keys are used for Diffie-Hellman key establishment for TLS communications.
FCS_CKM.2	The TOE supports Cryptographic Key Establishment using the following schemes: • RSA-based key establishment schemes that meets the following: NIST Special Publication 800-56B, “Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography” • The TOE implements RSA key establishment scheme with key sizes of 2048, and 3072 that is conformant to NIST SP800-56B. • Elliptic curve-based key establishment schemes that meets the following: NIST Special Publication 800-56A, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”

	ECDH public and private keys are used for Diffie-Hellman key establishment for TLS sessions. Please refer to Table#3 Cryptographic Algorithm Certificates for NIST CAVPs for RSA, ECDSA, and KAS/CVL ECC.
FCS_CKM_EXT.4	The TOE includes a Keychain Access program that allows users the ability to add, remove, and manage certificates and private keys. Keys stored in non-volatile memory are destroyed by a single overwrite consisting of zeros. Persistent keys are introduced into volatile memory after decryption or unwrapping and are also destroyed by a single overwrite consisting of zeroes. Ephemeral cryptographic keys are destroyed by a single overwrite consisting of zeroes.
FCS_COP.1(1)	The TOE supports AES encryption and decryption conforming to • CBC as specified in NIST SP 800-38A • GCM as specified in NIST SP 800-38D The AES key size supported are 128 bits and 256 bits and the AES modes supported are: CBC, and GCM. Please refer to Table#3 Cryptographic Algorithm Certificates for NIST CAVPs for AES.
FCS_COP.1(2)	The TOE supports Cryptographic hashing services conforming to FIPS PUB 180-4. The hashing algorithms are used for signature services and HMAC services. The following hashing algorithms supported: SHA-1, SHA-256, SHA-384 and SHA-512. The message digest sizes supported are: 160 bits, 256 bits, 384 bits and 512 bits. Please refer to Table #3 Cryptographic Algorithm Certificates for NIST CAVPs SHS.
FCS_COP.1(3)	The TOE provides Cryptographic signature generation and verification in accordance with the following cryptographic algorithms: • RSA digital signature algorithm conforming to FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 4. • The RSA key sizes supported are: 2048, and 3072. • Elliptical curve digital signature algorithm conforming to FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 5. • The Elliptical curve key size supported is 256 bits. Please refer to Table #3 Cryptographic Algorithm Certificates for NIST CAVPs for RSA and ECDSA.
FCS_COP.1(4)	The TOE supports Keyed-hash message authentication conforming to the Keyed-Hash Message Authentication Code and FIPS PUB 180-4 Secure Hash Standard with the following algorithms:

<table>
<tr><td></td><td>

Keyed hash algorithm authentication services in accordance with the following specified cryptographic algorithms: SHA-1, SHA-256, SHA-384 and SHA-512.
Key sizes supported are: 112 bits.

HMAC algorithms is used in support of TLS session.
<table>
<tr><th>HMAC Algorithms</th><th>Hash Functions</th><th>Block Size</th><th>Key lengths</th><th>MAC lengths</th></tr>
<tr><td>HMAC-SHA-1</td><td>SHA-1</td><td>512 bits</td><td>160 bits</td><td>160 bits</td></tr>
<tr><td>HMAC-SHA-256</td><td>SHA-256</td><td>512 bits</td><td>256 bits</td><td>256 bits</td></tr>
<tr><td>HMAC-SHA-384</td><td>SHA-384</td><td>1024 bits</td><td>384 bits</td><td>384 bits</td></tr>
<tr><td>HMAC-SHA-512</td><td>SHA-512</td><td>1024 bits</td><td>512 bits</td><td>512 bits</td></tr>
</table>
Please refer to Table #3 Cryptographic Algorithm Certificates for NIST CAVPs for HMAC.
</td></tr>
<tr><td>FCS_RBG_EXT.1</td><td>The TOE leverages CTR_DRBG (AES). The deterministic RBG used by the OS is seeded by an entropy source that accumulates entropy from a software-based noise source with a minimum of 256 bits of entropy at least equal to the greatest security strength (according to NIST SP 800-57) of the keys and hashes that it will generate.

Please refer to Table #3 Cryptographic Algorithm Certificates for NIST CAVPs for DRBG.</td></tr>
<tr><td>FCS_STO_EXT.1</td><td>The TOE stores the following sensitive data:

Usernames and passwords are used for authentication are maintained by whatever Directory Service / Credentials are configured
Trusted Certificates are used for establishing TLS sessions and are stored in the macOS Key chain.
Private Keys used for establishing TLS session and are stored in the macOS key chain.

macOS offers a repository, called Keychain, that conveniently and securely stores user names and passwords, including digital identities, encryption keys, and secure notes. It can be accessed by opening the Keychain Access app in /Applications/Utilities/. Using a keychain eliminates the requirement to enter—or even remember—the credentials for each resource. An initial default keychain is created for each Mac user, though users can create other keychains for specific purposes.

In addition to user keychains, macOS relies on a number of system-level keychains that maintain authentication assets that aren’t user-specific, such as network credentials and public key infrastructure (PKI) identities.</td></tr>
</table>

	Keychain items are encrypted using two different AES-256-GCM keys: a table key (metadata), and a per-row key (secret-key). Keychain metadata (all attributes other than kSecValue) is encrypted with the metadata key to speed searches while the secret value (kSecValueData) is encrypted with the secret-key. The meta-data key is protected by the Secure Enclave, but is cached in the application processor to allow fast queries of the keychain.
FCS_TLSC_EXT.1.1	The TOE implements TLS 1.2 (RFC 5246) supporting the following cipher suites: • TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246 • TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246 • TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246 • TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289 • TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289 • TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289 • TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289 The cipher suites specified are identical to those listed for this component.
FCS_TLSC_EXT.1.2	The macOS Catalina verifies that the presented identifier matches the reference identifier according to RFC 6125. The reference identifiers supported are DNS and IP addresses. The TOE does not support certificate pinning. Wild cards are supported.
FCS_TLSC_EXT.1.3	The macOS Catalina establishes a trusted channel if the peer certificate is valid.
FCS_TLSC_EXT.2.1	The TOE, by default, presents the Supported Elliptic Curves Extension in the Client Hello with the following NIST curves: secp256r1, secp384r1, and secp521r1.
FCS_TLSC_EXT.4.1	The TOE shall support mutual authentication using X.509v3 certificates.
FDP_ACF_EXT.1	The Apple File System (APFS) is the default file system for macOS Catalina and it provides access control to data in macOS Catalina. File system object attributes includes manipulation of metadata (e.g. change, access, modify time), as well as owner and permission data (e.g. group-ids for allowing multiple users to have the same access privileges, user-ids for individual access privileges, and permissions that can be assigned per user or group). These filesystem object attributes are based on the file system security schemes supported by macOS. macOS provides three file system security schemes: UNIX (BSD) permissions, POSIX access control lists (ACLs), and sandbox entitlements. In addition, the BSD layer provides several per-file flags that override UNIX permissions. These schemes are described in the sections that follow. • Unix (BSD) Permissions • POSIX access control lists • sandbox entitlements macOS allows admin users to disable ownership and permissions checking for removable volumes on a per-volume basis by choosing Get Info on the volume in Finder, then checking the ”Ignore ownership on this volume” checkbox.

	These permissions models fit together as follows: 1. If the app's sandbox forbids the requested access, the request is denied. 2. If ownership checking has been disabled for the volume in question by the system administrator (with a checkbox in its Finder Get Info window), the request is granted. 3. If an access control entry exists on the file, it is evaluated and used to determine access rights. 4. If a file flag prohibits the operation, the operation is denied. 5. Otherwise, if the user ID matches the owner of the file, the "user" permissions (also called "owner" permissions) are used. 6. Otherwise, if the group ID matches the group for the file, the "group" permissions are used. 7. Otherwise, the "other" permissions are used. Sandbox Entitlements: macOS supports the use of a sandbox to limit an app's ability to access files. These limits override any permissions the app might otherwise have. Sandbox limits are subtractive, not additive. Therefore, the file system permissions represent the maximum access an app might be allowed if its sandbox also permits that access. POSIX ACLs: The Mach and BSD permissions policies are supplemented by support in the kernel for ACLs (access control lists), which are data structures that provide much more detailed control over permissions than does BSD. For example, ACLs allow the system administrator to specify that a specific user can delete a file but cannot write to it. ACLs also provide compatibility with Active Directory and with the SMB/CIFS networks used by the Windows operating system. An ACL consists of an ordered list of ACEs (access control entries), each of which associates a user or group with a set of permissions and specifies whether each permission is allowed or denied. ACEs also include attributes related to inheritance. Unix Permissions Each file system object has a set of UNIX permissions defined by three attributes: • UID, short for user ID. Commonly referred to as the *File's Owner*. • GID, short for group ID. • Flags that include permission bits and other related attributes. The flags for a file or directory are a 16-bit value that is often represented as a three-digit or four-digit octal value (with the top four or seven bits dropped). The Owner, Group, and Other bit sets contain three bits: read, write, execute (rwx for short). BSD File Flags

	In addition to the standard UNIX file permissions, macOS supports several BSD file flags provided by the chflags API and the related chflags command. These flags override the UNIX permissions. Each ACL on a directory can contain any combination of the following inheritance flags: • Inherited (this ACE was inherited) • File Inherit (this ACE should be inherited by files created within this directory) • Directory Inherit (this ACE should be inherited by directories created within this directory) • Inherit Only (this ACE should not be checked during authorization) • No Propagate Inherit (this ACE should be inherited only by direct children; that is, the ACE should lose any Directory Inherit or File Inherit bit when inherited) When it creates a new file, the kernel goes through the entire access control list of the parent directory and copies to the file's ACL any ACEs that are marked for file inheritance. Similarly, when it creates a new subdirectory, the kernel copies to the subdirectory's ACL any ACEs that are marked for directory inheritance. If a file is copied and pasted into a directory, the kernel replicates the contents of the source file into a new file at the destination. Because it is creating a new file, the system checks the ACL of the parent directory and adds any inherited ACEs to whatever ACEs were in the original file. If a file is moved into a directory, on the other hand, the original file is not replicated and no ACEs are inherited. In this case, the parent directory's ACEs are added to the moved file only if the administrator specifically propagates ACEs from the parent directory through contained files and subdirectories. Similarly, once a file has been created, changing the ACL of the parent directory does not affect the ACL of contained files and subdirectories unless the administrator specifically propagates the change. In BSD, applying a directory's permissions to enclosed files and subdirectories completely replaces the permissions of the enclosed objects. With ACLs, in contrast, inherited ACEs are added to other ACEs already on the file or directory. The order in which ACEs are placed in an ACL—and therefore the order in which they are evaluated to determine permissions—is as follows: 1. Explicitly specified deny associations 2. Explicitly specified allow associations Inherited associations, in the same order in which they appeared in the parent. Since ACEs can be inherited, administrators can control the fine-grained permissions of files created in a directory by assigning inheritable ACEs to the directory. Doing so saves the work of assigning ACEs to each file individually. In addition, because ACEs can apply to groups of users, administrators can assign permissions to groups rather than having to specify permissions for each

	individual. Applying access security to directories and groups rather than to files and individuals saves administrator time and gives better file system performance in many circumstances.
FIA_AFL.1	The TOE will detect when an administrator configurable integer within 1-50 unsuccessful authentication attempts for authentication based on user name and password attempts have been met. Once the specified number of unsuccessful authentication attempts for an account has been met, the OS will lock out the account.
FIA_UAU.5	The TOE supports authentication based on username and password and smart cards. For password-based authentication, the user account contains a username and a password. A random salt is created for the password in a Password-Based Derivation Key Function 2 (PBKDF2) with SHA-512. This result is then stored in the Directory Services node. When a user logs into the system, the TOE uses the entered password and the randomly generated salt and compares this with the stored value. If they match, then the user is granted access to the system. If the values do not match, then the user is not granted access. Smart card authentication provides a strong two-factor authentication in macOS Catalina. This requires the user to have a username and a PIN. The user initially logins in providing a valid username and password. Once successfully authenticated, a smart card is paired to the user account. When the smart card pairing is initiated, the user is required to enter the smart card's PIN to unlock the card. The PIN is not stored by the TOE. The TOE then passes the entered PIN to the smart card for verification. Upon successful verification, the smart card's certificate (which contains its public key) is sent to the TOE for storage. The certificate is associated with the user's account and the card is considered to be paired with the user. When a user inserts a Smart Card into the host platform, the user enters the associated PIN to unlock the card. Once unlocked, a signing operation is performed by the card. The TOE verifies the signature using the paired certificate for authentication.
FIA_X509_EXT.1	When an X.509 certificate is presented, the TOE verifies the certificate path, and certification validation process by verifying the following rules: • RFC 5280 certificate validation and certificate path validation • The certificate path must terminate with a trusted CA certificate • The OS shall validate a certificate path by ensuring the presence of the basicConstraints extension, that the CA flag is set to TRUE for all CA certificates, and that any path constraints are met. • The TSF shall validate that any CA certificate includes caSigning purpose in the key usage field

	The OS shall validate the extendedKeyUsage field according to the following rules: • Certificates used for trusted updates and executable code integrity verification shall have the Code Signing Purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field. • Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field. • Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the EKU field. • S/MIME certificates presented for email encryption and signature shall have the Email Protection purpose (id-kp 4 with OID 1.3.6.1.5.5.7.3.4) in the EKU field. • OCSP certificates presented for OCSP responses shall have the OCSP Signing Purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the EKU field. • Server certificates presented for EST shall have the CMC Registration Authority (RA) purpose (id-kp-cmcRA with OID 1.3.6.1.5.5.7.3.28) in the EKU field. (conditional) X509 certificates are validated when imported into the TOE's trusted certificate store, during session establishment with a peer and prior to presenting a certificate to the peer during trusted channel implementation using TLS for mutual authentications.
FIA_X509_EXT.1.2	The OS shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.
FIA_X509_EXT.2	The TOE uses X.509v3 certificates for performing mutual authentication for TLS in HTTPS connections.
FMT_MOF_EXT.1	The TOE supports the following roles: Administrator and User. The Administrator is a member of the local admin group whereas the User is not a member of the local group. The Administrator has access to the following management functions: • Enable/disable screen lock • Configure screen lock inactivity timeout • Configure local audit storage capacity • Configure minimum password Length • Configure minimum number of special characters in password • Configure minimum number of numeric characters in password • Configure minimum number of uppercase characters in password • Configure minimum number of lowercase characters in password • Configure lockout policy for unsuccessful authentication attempts through [limiting number of attempts during a time period] • Configure host-based firewall • Configure name/address of directory server with which to bind • Configure name/address of remote management server from which to receive management settings

<table>
<tr><td></td><td>

Configure name/address of audit/logging server to which to send audit/logging records
Configure audit rules
Configure name/address of network time server
Enable/disable automatic software update
Configure WiFi interface
Enable/disable Bluetooth interface

The user has access to the following management functions:

Enable/disable screen lock
Configure screen lock inactivity timeout
Enable/disable Bluetooth interface

</td></tr>
<tr><td>FMT_SMF_EXT.1</td><td>
The TOE maintains the following roles: Administrator and User.

The management functions are listed below:
<table>
<tr><th>Management Function</th><th>Administrator</th><th>User</th></tr>
<tr><td>Enable/disable [screen lock]</td><td>X</td><td>X</td></tr>
<tr><td>Configure [screen lock] inactivity timeout</td><td>X</td><td>X</td></tr>
<tr><td>Configure local audit storage capacity</td><td>X</td><td>-</td></tr>
<tr><td>Configure minimum password Length</td><td>X</td><td>-</td></tr>
<tr><td>Configure minimum number of special characters in password</td><td>X</td><td>-</td></tr>
<tr><td>Configure minimum number of numeric characters in password</td><td>X</td><td>-</td></tr>
<tr><td>Configure minimum number of uppercase characters in password</td><td>X</td><td>-</td></tr>
<tr><td>Configure minimum number of lowercase characters in password</td><td>X</td><td>-</td></tr>
<tr><td>Configure lockout policy for unsuccessful authentication attempts through [limiting number of attempts during a time period]</td><td>X</td><td>-</td></tr>
<tr><td>Configure host-based firewall</td><td>X</td><td>-</td></tr>
<tr><td>Configure name/address of directory server with which to bind</td><td>-</td><td>-</td></tr>
<tr><td>Configure name/address of remote management server from which to receive management settings</td><td>-</td><td>-</td></tr>
<tr><td>Configure name/address of audit/logging server to which to send audit/logging records</td><td>X</td><td>-</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td>
<table>
<tr><td>Configure audit rules</td><td>X</td><td>-</td></tr>
<tr><td>Configure name/address of network time server</td><td>X</td><td>-</td></tr>
<tr><td>Enable/disable automatic software update</td><td>X</td><td>-</td></tr>
<tr><td>Configure WiFi interface</td><td>X</td><td>-</td></tr>
<tr><td>Enable/disable Bluetooth interface</td><td>X</td><td>X</td></tr>
<tr><td>Enable/disable [no other external interfaces]</td><td>-</td><td>-</td></tr>
<tr><td>[no other management functions]</td><td>-</td><td>-</td></tr>
</table>
</td></tr>
<tr><td>FPT_ACF_EXT.1</td><td>The TOE provides access control policy through the system integrity protection. This technology prevents from malicious software from modifying files and folders. The System Integrity program restricts the root user account (an administrator superuser account) and limits the actions that the root user can perform on protected parts of the Mac operating system.

System Integrity Protection includes protection for these parts of the system:
/var
Apps that are pre-installed with macOS Catalina
Kernel drivers and modules:
-/System/Library/Extensions/
Security audit logs:
-/var/audit/*
Shared libraries:
-/Library/Frameworks/
-/System/Library/Frameworks/
-/System/Library/PrivateFrameworks/
System executables:
/System
/usr
/bin
/sbin
-/Applications
System configuration files:
-System-wide “preferences”
-/Library/Preferences/
-User-specific “preferences”
Security Audit Logs:
-/etc/security/audit-control
System-wide local directory services credentials:
-/private/var/db/dslocal/nodes/Default/</td></tr>
</table>

	System Integrity Protection is designed to allow modification of these protected parts only by processes that are signed by Apple and have special entitlements to write to system files, such as Apple software updates and Apple installers. Apps that you download from the Mac App Store already work with System Integrity Protection. System Integrity Protection also helps prevent software from selecting a startup disk. To select a startup disk, choose System Preferences from the Apple menu, then click Startup Disk. Or hold down the Option key while you restart, then choose from the list of startup disks.
FPT_ASLR_EXT.1	The TOE always randomizes process address memory locations with 16 bits of entropy.
FPT_SBOP_EXT.1	The macOS Catalina employs stack-based buffer overflow protections using address space layout randomization and non-executable stack and heap. The host platforms of the macOS Catalina support a feature called the NX bit which 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 will crash. The macOS Catalina takes leverages this feature by marking the stack and heap as non-executable. This makes buffer overflow attacks difficult because any attacks that places executable code on the stack or heap and then tries to execute that code will fail. The rational for all the binaries which are not protected by SBOP are the following: Type 1: The compiler can optimize away stack usage (which is certainly something macOS heavily rely on for performance reasons). Type 2: Some binaries are just small entry points that rely on system frameworks for all of their functionality. There, the binary itself is going to be really small (less than ~1000 instructions, sometimes as small as 10 instructions), so is much less likely to need stack protection. Type 3: There are very short program/functions that does not access the stack (and just forwards to system frameworks to do the real work) Type 4: There are tiny binaries with a single trivial function that does not need stack protections or tiny wrappers that does not make use of the stack. Type 5: Some binaries do not access the stack in any kind of vulnerable way. The TOE also randomizes process address memory location with 16 bit of entropy.

FPT_TST_EXT.1	When the OS boots with the T2 chip on, the chip executes code from read-only memory known as the Boot ROM. This immutable code, referred to as the hardware root of trust, is laid down during chip fabrication and is audited for vulnerabilities and implicitly trusted. The Boot ROM code contains the Apple Root CA public key, which is used to verify that the iBoot bootloader is signed by Apple's private key before allowing it to load. This is the first step in the chain of trust. iBoot verifies the kernel and kernel extension code on the T2 chip, which subsequently verifies the Intel UEFI firmware. The UEFI firmware and the associated signature are initially available only to the T2 chip. After verification, the UEFI firmware image is mapped into a portion of the T2 chip memory and this memory is made available to the (Intel) application processor via the enhanced Serial Peripheral Interface (eSPI). When the application processor first boots, it fetches the UEFI firmware via eSPI from the integrity-checked, memory-mapped copy of the firmware located on the T2 chip. The evaluation of the chain of trust continues on the application processor, with the UEFI firmware evaluating the signature for boot.efi, which is the macOS bootloader. The Intel-resident macOS secure boot signatures are stored in the same Image4 format used for iOS and T2 chip secure boot, and the code that parses the Image4 files is the same hardened code from the current iOS secure boot implementation. Boot.efi in turn verifies the signature of a new file called immutablekernel. When secure boot is enabled, the immutablekernel represents the complete set of Apple kernel extensions required to boot macOS. The secure boot policy terminates at the handoff to the immutablekernel, and after that, macOS security policies (such as System Integrity Protection and signed kernel extensions) take effect. Any errors or failures in this process result in Mac entering macOS Recovery mode, Apple T2 Security Chip recovery mode, or Apple T2 Security Chip DFU mode.
FPT_TUD_EXT.1 FPT_TUD_EXT.2	The Mac operating system and software application updates can be downloaded manually through the following website: • https://support.apple.com/ When an update is initiated, the TOE downloads the update package and performs the RSA 2048-bit digital signature verification. If the verification is successful, the TOE installs the update and reboots the host device. If the verification is unsuccessful, the TOE terminates the updates process.
FTP_ITC_EXT.1	The TOE uses TLS as conforming to FCS_TLSC_EXT.1.
FTP_TRP.1	The TOE provides a trusted path between itself and local users only using TLS v1.2 protocol.
FTA_TAB.1	The TOE will display an advisory warning message regarding unauthorized use of the OS prior to establishing a user session.

Table 17 TOE Summary Specification SFR Description

8 Annex A: References

Identifiers	Descriptions
[CC_PART1]	Common Criteria for Information Technology Security Evaluation – Part 1: Introduction and general model, dated April 2017, version 3.1, Revision 5, CCMB-2017-04-001
[CC_PART2]	Common Criteria for Information Technology Security Evaluation – Part 2: Security functional components, dated April 2017, version 3.1, Revision 5, CCMB-2017-04-002
[CC_PART3]	Common Criteria for Information Technology Security Evaluation – Part 3: Security assurance components, dated April 2017, version 3.1, Revision 5, CCMB-2017-04-003
[CEM]	Common Methodology for Information Technology Security Evaluation – Evaluation Methodology, dated September 2012, version 3.1, Revision 5, CCMB-2017-04-004
[800-38A]	NIST Special Publication 800-38A Recommendation for Block 2001 Edition Recommendation for Block Cipher Modes of Operation Methods and Techniques December 2001
[800-56A]	NIST Special Publication 800-56A Rev 2, May 2013
[800-56B]	NIST Special Publication 800-56B Recommendation for Pair-Wise, August 2009
[800-38A]	[NIST Special Publication 800-38A Recommendation for Block 2001 Edition Recommendation for Block Cipher Modes of Operation Methods and Techniques December 2001
[800-38D]	NIST Special Publication 800-38D Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC, November 2007

Table 18 Annex A: References